

# The Command Line in Windows

Vic Laurie

All rights reserved, Copyright 2005-2010

# The Command Line in Windows

## The Windows Command Line, Batch Files, and Scripting

The Windows command line is a mainstay for systems administrators and power users but is relatively unknown to many PC users. The purpose of this site is to make the power and utility of the command line more familiar to a wider community of computer users. Also under-appreciated are the related resources of batch files and scripts and these will be discussed as well.

Windows is famous as a graphical user interface and many computer users tend to forget (or never knew) that there are also very useful command line functions in Windows. In fact, there is a great deal more to Windows than just point and click. The keyboard and the command line can be substantial adjuncts to the mouse and icons. There are two basic features involving a command line. One is the entry 'Run" (or "Start Search" in Vista) that is in the Start menu and the other is the command prompt window.

### Introduction to the Command Line

Those who are new to the command line or need a refresher can read basic material on these pages:

- [Command Line- Introduction](#)
- [Command line list and reference](#)
- [Commands that everybody can use](#)
- [Configuring the command prompt window](#)
- [Start-Run line](#)

### Specific Applications of the Command Shell

Some details and examples for various commands are considered in a series of pages listed below. The subjects include computer maintenance, system administration, file management, Internet tools, and network administration.

#### File management

- [Assoc](#)
- [Ftype](#)
- [Xcopy](#)

#### Network and Internet tools

- [Net Services \(Net\)](#)
- [Netstat](#)
- [Network Services Shell \(Netsh\)](#)
- [TCP/IP networking tools](#)

## Site navigation

powered by [FreeFind](#)

## Pages

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the](#)

[command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

## System administration and maintenance

- [File system utility- Fsutil](#)
- [Recovery Console](#)
- [Recovery Console- Commands](#)
- [Registry editor console](#)
- [Service Controller Command \(SC\)](#)
- [Tasklist](#)
- [Taskkill](#)
- [Tskill](#)
- [Xcopy](#)

## Additions and extensions to native commands

- [Scripts in the command line](#)
- [Server 2003 tools for XP](#)
- [Support tools](#)

## Batch files

Batch files provide a simple way to perform many repetitive or time-consuming tasks. While batch files can be quite sophisticated, the basics are simple enough to be useful to the average PC user with no knowledge of programming.

- [Introduction to Batch files](#)
- [Branching and Looping with "If: and "Goto"](#)
- [Iterating and Looping with "For... in...do"](#)
- [Variables and the "Set" command](#)

## Other Command Line Topics

- [Doskey](#)
- [PowerShell](#)
- [Tips for using the command shell](#)

## Vista

- [Shell command](#)
- [Vista command list](#)
- [Vista command line tips](#)

# The Command Line in Windows

## The Windows Command Line, Batch Files, and Scripting

The Windows command line is a mainstay for systems administrators and power users but is relatively unknown to many PC users. The purpose of this site is to make the power and utility of the command line more familiar to a wider community of computer users. Also under-appreciated are the related resources of batch files and scripts and these will be discussed as well.

Windows is famous as a graphical user interface and many computer users tend to forget (or never knew) that there are also very useful command line functions in Windows. In fact, there is a great deal more to Windows than just point and click. The keyboard and the command line can be substantial adjuncts to the mouse and icons. There are two basic features involving a command line. One is the entry 'Run" (or "Start Search" in Vista) that is in the Start menu and the other is the command prompt window.

### Introduction to the Command Line

Those who are new to the command line or need a refresher can read basic material on these pages:

- [Command Line- Introduction](#)
- [Command line list and reference](#)
- [Commands that everybody can use](#)
- [Configuring the command prompt window](#)
- [Start-Run line](#)

### Specific Applications of the Command Shell

Some details and examples for various commands are considered in a series of pages listed below. The subjects include computer maintenance, system administration, file management, Internet tools, and network administration.

#### File management

- [Assoc](#)
- [Ftype](#)
- [Xcopy](#)

#### Network and Internet tools

- [Net Services \(Net\)](#)
- [Netstat](#)
- [Network Services Shell \(Netsh\)](#)
- [TCP/IP networking tools](#)

## Site navigation

powered by [FreeFind](#)

## Pages

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the](#)

[command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

## System administration and maintenance

- [File system utility- Fsutil](#)
- [Recovery Console](#)
- [Recovery Console- Commands](#)
- [Registry editor console](#)
- [Service Controller Command \(SC\)](#)
- [Tasklist](#)
- [Taskkill](#)
- [Tskill](#)
- [Xcopy](#)

## Additions and extensions to native commands

- [Scripts in the command line](#)
- [Server 2003 tools for XP](#)
- [Support tools](#)

## Batch files

Batch files provide a simple way to perform many repetitive or time-consuming tasks. While batch files can be quite sophisticated, the basics are simple enough to be useful to the average PC user with no knowledge of programming.

- [Introduction to Batch files](#)
- [Branching and Looping with "If: and "Goto"](#)
- [Iterating and Looping with "For... in...do"](#)
- [Variables and the "Set" command](#)

## Other Command Line Topics

- [Doskey](#)
- [PowerShell](#)
- [Tips for using the command shell](#)

## Vista

- [Shell command](#)
- [Vista command list](#)
- [Vista command line tips](#)

# The Command Line in Windows

## Managing Files from the Command Line- Assoc and Ftype

Windows comes with several command-line tools for file management. The features and applications of Assoc and Ftype are discussed.

Before discussing the file management tools, I would like to quickly review some of the basic facts about how Windows manages files. (More details can be found at [another site](#).) A very basic property of a file is its *file type*. Each file type has a set of specific actions that can be carried out with it or to it. The software that is assigned to do these actions with or to a particular file type is said to be "associated" with the file type. There may be several actions and different software may be involved for each particular action. This set of software constitutes the *program associations* for a given file type. The *extension* of a file is a tag that tells the computer what the file type is and what is to be done with the file when it is opened or double-clicked or otherwise invoked one way or the other. Microsoft also uses the word "associate" in connection with file extensions and refers to an extension being associated with a particular file type. All of this information is stored in the Registry and can be edited or changed in several ways. I have discussed methods that use the graphical interface on [another site](#). Here we look at using the command line.

### Manage file type and extension associations with the "assoc" command

This tool is very useful for managing the relationship or association between file extensions and file types. the syntax is

```
assoc [.ext[=[fileType]]]
```

If the plain command "assoc" is entered, you will get a list of what file types correspond to the extensions currently registered on the system. The list can be quite long so it is best to redirect to a file or to pipe to the "more" command so that one full screen at a time can be viewed

```
assoc > list.txt OR assoc | more
```

If the only parameter is a file extension (including the leading period), the file type for that extension will be given. For example, to see what file type is associated with *.txt*, enter

```
assoc .txt
```

Since the names used for the various file types may not always be obvious, the above can be a useful type of command, In this case, the output would normally be

```
.txt=txtfile
```

.. To delete the file type association for the file name extension *.xyz* (use with care), enter

```
assoc .xyz=
```

Another use is to associate a given extension with a certain file type. As an example, to associate the extension *.log* with type *txtfile*, enter

```
assoc .log=txtfile
```

Note that more than one extension can be associated with a file type. For example, the file type "jpegfile" typically has both the extensions *.jpg* and *.jpeg* associated with it. Also note that it is possible to create your own file extensions and to associate them with a filetype.

For a Microsoft reference on assoc [go here](#) or to the Windows Help and Support Center. You can also enter the command

```
assoc /?
```

### Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console- Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the](#)

[command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)[Back to top](#)

## Manage file type and program associations with the "ftype" command

As previously mentioned, each file type has a set of operations and corresponding software associated with it. In particular, all active file types have an action named "Open" that is the default action. This is the action that is invoked when you double-click a file with an extension associated with the file type. There may also be other actions (listed in the Context Menu) but *ftype* deals with "Open". The "Open" action is defined by a string that includes the fully qualified path to the executable file that is to carry out the action and any parameters that must be passed to the executable. The syntax for *ftype* is

```
ftype [fileType[=[openCommandString]]]
```

Entering the bare command "ftype" will list all of the current file types that have the open command strings defined and the corresponding command string. It can be quite a long list so it is best to redirect to a file or to pipe to the "more" command. Having the list can be convenient as a record of what programs are being used to open various files.

If a particular file type is specified, then the command string for that file type will be displayed. Using the text file type as an example, you would enter

```
ftype txtfile
```

This would produce the output

```
txtfile=%SystemRoot%\system32\notepad.exe %1
```

This shows that the executable file that opens text files is *notepad.exe* located in the folder `Windows\system32\` (The [environment variable](#) `%SystemRoot%` is used to indicate the Windows folder.) Note the presence of the placeholder `%1`. This is necessary because the full command for the open action requires the name of the file that is to be opened, and the placeholder stands for the file name. This command is useful when you want to see what program opens a particular file type.

If it were desired to change the *openCommandString* to use Wordpad instead of Notepad, the command (on one of my computers) would be

```
ftype txtfile="G:\Program Files\Windows NT\Accessories\wordpad.exe" "%1"
```

(The path for Wordpad will vary from one computer to the next. This example is for illustration only). Note the use of quotation marks to enclose a path with spaces in it. While changing program associations may be easier using the Windows Explorer Tools-Folder Options dialog (no typing required), the command line method can also be useful, especially in batch files.

[Back to top](#)



# The Command Line in Windows

## Site navigation

powered by [FreeFind](#)

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console- Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

## Batch Files (Scripts) in Windows

Batch files or scripts are small easy-to-write text files that carry out a series of commands. They can be simple enough that even the average home computer user can take advantage of them.

Systems administrators and power users are well aware of the utility of batch files but the average PC user is generally unacquainted with them or is intimidated by the notion of writing or even running a script. This is unfortunate since it means that many are foregoing the use of a powerful tool for carrying out routine or repetitive tasks. Although batch files can be quite sophisticated and used for complicated network and system administration, they can also be of the utmost simplicity and very brief. In this article, I will introduce the batch file and discuss some uncomplicated examples that make basic tasks easier.

### What is a batch file?

These are simple text files containing some lines with commands that get executed in sequence, one after the other. These files have the special extension BAT or CMD. Files of this type are recognized and executed through an interface (sometimes called a shell) provided by a system file called the command interpreter. In Windows XP/ Vista the command interpreter is the file *cmd.exe*. The large assortment of versatile commands available in Windows XP/Vista makes batch files a powerful tool.

Constructing a batch file consists of nothing more than opening any text editor like the accessory *Notepad*, entering some lines containing commands, and saving the file with an extension BAT or CMD. (The CMD extension is limited to newer Windows systems and is not recognized in Windows 9x/Me systems. In Windows XP and Vista, there is little practical difference between the two extensions.) Don't use *Wordpad* or *Word* unless you are very careful to save all files in pure text format. The commands themselves are often quite simple and there is no need to learn a programming language. Those who wish can explore the intricacies that are available with [branching](#) and [looping](#) but here I will confine the discussion to some straightforward application to everyday tasks. The focus will be on saving time and effort for some routine stuff like system housekeeping and simple file management.

Running a batch file is a simple matter of clicking on it. Batch files can also be run in a command prompt or the Start-Run line. In that case, the full path name must be used unless the file's path is in the [path environment](#).

### Constructing a batch file

In the following discussion it is assumed that the [Introductory](#) page and the page on [Commands](#) have been read.

The first line in a batch file often consists of this command

```
@echo off
```

By default, a batch file will display its commands as it runs. The purpose of this first command is to turn off this display. The command "echo off" turns off the display for the whole script, except for the "echo off" command itself. The "at" sign "@" in front makes the command apply to itself as well. This nuance isn't really all that important in the context here but I mention it because it is often seen in scripts. The scripts we will discuss are very brief and omitting this line won't make any great difference.

However, as a matter of good practice, we will enter it in our scripts.

Our first batch file example is going to list all the files in a folder and put the list in a new text file . We will use the directory command "dir" that is [discussed on another page](#). Open Notepad and enter the line "@echo off" (without quotes). Next enter



[Tips for using the command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

another line

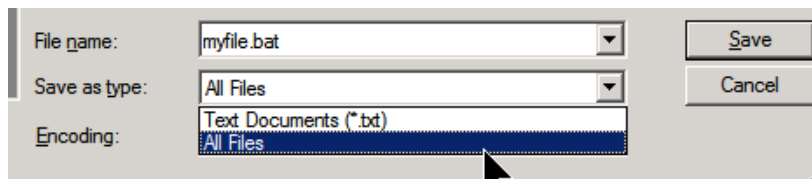
```
dir "C:\Program Files" > C:\list_of_program_files.txt
```

(I'm assuming that your Program Files folder is on the C: drive). This will give us the two-line file

```
@echo off
```

```
dir "C:\Program Files" > C:\list_of_program_files.txt
```

Now save this two-line file as "listprograms.bat" (without quotes) to some convenient location. Be sure that Notepad is saving as "All files" and not as a text file. See the figure below.



Three important points are illustrated in the example script. Note that complete paths are used for files including the drive letter. Also note the quotes around "C:\Program Files". Paths must be quoted whenever a file or folder name has a space in it. Finally note the [redirection symbol](#) ">" that is used to send the output to a file instead of the screen.

All that has to be done to use the file is to double-click it. A file *C:\list\_of\_program\_files.txt* will then be created.

## A more general version with arguments

The file that we have been discussing is limited to listing one particular folder and putting the list in one particular file. However, it is easy to make the file able to list whatever folder we want and to put the list wherever we want. Batch files can use arguments or data that is input from the user. The process makes use of placeholders of the form %1, %2. These are replaced in the script by our input data. This type of situation cannot be clicked directly but should be run in a command prompt. The new batch file would be

```
@echo off
dir %1 > %2
```

Enter in Notepad and save as "makelist.bat". To run the file, open a command prompt and enter

```
{path}makelist somefolder somewhere\list.txt
```

where *somefolder* is whatever folder (with complete path) that you want to list in *somewhere\list.txt*. Now you have a little program that will list the contents of a folder whenever you want. If you want a list of all the subfolders as well, use the command

```
dir /s %1 > %2
```

If you want a list that only includes files of a certain type, MP3 files for example, use

```
dir %1\*.mp3 > %2
```

The line above illustrates the use of the wildcard "\*". The ability to use wildcards greatly enhances the power of batch files.

Life will be easier if you put all batch scripts in a folder that is in the [path environment](#).

## The Rem statement

Very often batch files contain lines that start with "Rem". This is a way to enter comments and documentation. The computer ignores anything on a line following Rem. For batch files of any complexity, comments are a good idea. Note that the command interpreter actually reads Rem statements so using too many can slow down execution of a script.

## More examples

Following the discussion on [another page](#), it is easy to create batch files for some typical maintenance. To create a very simple backup script, use [xcopy](#). The code might be

```
xcopy %1 %2 /d /s
```

This will update all files in the input source folder %1 and its subfolders by copying to the backup folder %2. In practice, a useful backup script would probably need a few more of the switches discussed at [xcopy](#).

Again following [previous discussion](#) of the command "del", a file to delete all temporary files with extension TMP might contain

```
del %1\*.tmp
```

## Prompting for user input

You can also interact with a user and ask that data be entered. The old DOS had a "Choice" command for very limited interaction but that has been superseded in Windows XP/Vista by the more versatile "set /p". The syntax is:

```
set /p variable= [string]
```

"Variable" is the name of the variable that will be assigned to the data that you want the user to input. "String" is the message that the user will see as a prompt. If desired, "string" can be omitted. Here is an example that asks the user to enter his or her name:

```
set /p name= What is your name?
```

This will create a variable %name% whose value is whatever the user enters. Note that the user must press the "Enter" key after typing the input.

(The "Choice" command has returned as a more powerful version in Vista.)

## Further reading

These are simple examples and this page does not pretend to explain everything about batch files. The idea is to show how simple they are and to intrigue readers to look further into the subject. Even more powerful batch files can be constructed with the addition of simple decision making and methods of doing the same thing many times. Branching with "If" and "Goto" are [discussed next](#) ; using "For" [to do repetitive tasks](#) is considered on a third page.

Batch files are discussed in many books on Windows, at numerous Web sites and at this [Microsoft site](#). Even if you do not want to write them, there are many already available for your use. This [page at a sister site](#) lists a number of sources.

[Back to top](#)

Next: [Branching and looping](#)

# The Command Line in Windows

## More Powerful Batch Files Part I - Branching and Looping

### Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console- Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the command shell](#)

The commands "If...else" and "goto" are discussed.

Although many useful tasks can be carried out with simple batch files containing just a few lines, the full power that is available can only be realized with the more advanced methods of branching, iterating, and looping. These methods are among the tools used by programmers to create very sophisticated scripts. However, the concepts are actually quite easy to grasp and are accessible to those with no background in programming. Just a few extra lines in a batch file using these tools can add a very significant increase in versatility and power. On this page, I will discuss branching. [In part two of more advanced methods](#) I will introduce iterative methods.

### Conditional branching with "If" statements

Batch files can make decisions and choose actions that depend on conditions. This type of action makes use of a statement beginning with "If". The basic meaning of an "If" statement is

```
If something is true then do an action (otherwise do a different action)
```

The second part of the statement (in parentheses) is optional. Otherwise, the system just goes to the next line in the batch file if the first condition isn't met. The actual syntax is

```
If (condition) (command1) Else (command2)
```

The "Else" part is optional. The form "If not" can also be used to test if a condition is false. Note that "If" tests for true or false in the Boolean sense.

#### "If exist" statement

There is a special "If exist" statement that can be used to test for the existence of a file, followed by a command. An example would be:

```
If exist somefile.ext del somefile.ext
```

You can also use a negative existence test:

```
if not exist somefile.ext echo no file
```

#### "If defined" statement

Another special case is "if defined ", which is used to test for the existence of a variable. For example:

```
if defined somevariable somecommand
```

This can also be used in the negative form, "if not defined".

#### "If errorlevel" statement

Yet another special case is "if errorlevel", which is used to test the exit codes of the last command that was run. Various commands issue integer exit codes to denote the status of the command. Generally, commands pass 0 if the command was completed successfully and 1 if the command failed. Some commands can pass additional code values. For example, there are

[Tskill and Taskkill](#)[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

five exit code values for [Xcopy](#). These exit code values are stored in the special variable `errorlevel`. An example command would be:

```
if errorlevel n somecommand
```

where "n" is one of the integer exit codes. Note that the comparison is done by checking if `errorlevel` is greater than or equal to `n`. If used with "not" the comparison checks if `errorlevel` is less than `n`.

## Comparison operators

In some cases the condition to be met is obtained by comparing strings. For example

```
if string1 == string2
```

Note that the "equals" sign is written twice. This condition is met if the two strings are exactly identical, including case. To render comparisons insensitive to case, use the switch `/i`. For more general comparisons, use the operators in Table I. (The operators are given in upper case in the table but they are not case-dependent.) Numerical comparisons only work with all-digit strings. Otherwise, the comparison is done alphabetically. For example "a" is less than "b". For case independence, use the switch `/i`. An example command might read:

```
if /i string1 gtr string2 somecommand
```

When comparing variables that are strings, it may be best to enclose the variable name in quotes. For example, use:

```
if "%1" == somestring somecommand
```

## The "goto" command

Generally, the execution of a batch file proceeds line-by-line with the command(s) on each line being run in turn. However, it is often desirable to execute a particular section of a batch file while skipping over other parts. The capability to hop to a particular section is provided by the appropriately named "goto" command (written as one word). The target section is labeled with a line at the beginning that has a name with a leading colon. Thus the script looks like

```
...
goto :label
...some commands
:label
...some other commands
```

Execution will skip over "some commands" and start with "some other commands". The label can be a line anywhere in the script, including before the "goto" command.

"Goto" commands often occur in "if" statements. For example you might have a command of the type:

```
if (condition) goto :label
```

## The "End of File" (:eof) label for exiting a script

Sometimes it is desirable to terminate a script if a certain condition is met (or not met). One way to exit is to use the special label `:eof` in a `goto` command. The label is not actually placed in the batch file. Windows XP and later recognize `:eof` without any label explicitly placed at the end of the batch file. Thus if you need to test for a particular condition that makes script termination desirable, you can write:

```
if (condition) goto :eof
```

Note that this terminates the script but does not necessarily close the command shell.

Table I. Comparison operators in "If" statements

Operator	Meaning
EQU	equal to
NEQ	not equal to
LSS	less than
LEQ	less than or equal to
GTR	greater than
GEQ	greater than or equal to

## Looping with "If" and "Goto"

An ancient method for doing repetitive tasks uses a counter, "if" statements, and the "goto" command. The counter determines how many times the task is repeated, the "if" statement determines when the desired number of repetitions has been reached, and the "goto" command allows for an appropriate action to be executed, either the repetitive task or exiting. Generally, the more elegant methods provided by the powerful "for...in...do" command are preferable and they are discussed [on the next page](#). However, for completeness and to illustrate some of what we have discussed, I will give an example that uses the clumsier method.

The simple script below creates the numbers 1 to 99 and sends them to a file. It uses the ["set" command](#) to create a variable that is also the counter for how many times we have iterated.

```
@echo off
set /a counter=0
:numbers
set /a counter=%counter%+1
if %counter% ==100 (goto :eof) else (echo %counter% >> E:\count.txt)
goto :numbers
```

(Best programming practice would dictate that the variable %counter% be localized or destroyed at the end but for simplicity I have omitted the several extra lines needed to do that. As written, this environment variable would persist until the command shell itself, not just the script, was closed.)

In anticipation, I can note that the same result as the script above can be achieved with a two-line script using the "for" statement discussed [on the next page](#):

```
@echo off
for /l %%X in (1,1,99) do (echo %%X >> E:\count.txt)
```

[Back to top](#)

Next: [Iteration methods](#)

# The Command Line in Windows

## More Powerful Batch Files Part II - Iterating with "For"

### Site navigation

[Home](#)
[Assoc and Ftype](#)
[Batch files- basics](#)
[Batch files - branching](#)
[Batch files- iterating](#)
[Command Line-](#)
[Introduction](#)
[Command line list and](#)
[reference](#)
[Commands that everybody  
can use](#)
[Configuring the command  
prompt window](#)
[Doskey](#)
[File system utility- Fsutil](#)
[Net Services](#)
[Netstat](#)
[Network Services Shell](#)
[PowerShell](#)
[Recovery Console](#)
[Recovery Console-  
Commands](#)
[Registry editor console](#)
[Scripts in the command  
line](#)
[Server 2003 tools for XP](#)
[Service Controller](#)
[Command \(SC\)](#)
[Shell command](#)
[Start-Run line](#)
[Support tools](#)
[Tasklist](#)
[TCP/IP networking tools](#)
[Tips for using the](#)
[command shell](#)
[Tskill and Taskkill](#)

The very useful "for...in...do" statement is discussed

Computers are very good at doing the same thing over and over. The command line contains a powerful and versatile method for carrying out this type of operation. With this method, you can automate many time-consuming tasks. The basic statement is of the form:

```
for {each item} in {a collection of items} do {command}
```

(For those who persist in calling the command line DOS, note that the 32-bit version of the "For" statement is much more powerful than the old 16-bit DOS version.)

A single-letter replaceable variable is used to represent each item as the command steps through the the collection (called a "set"). Note that, unlike most of Windows, variables are case-dependent. Thus "a" and "A" are two different variables. The variable has no significance outside the "For" statement. I will be using X throughout the discussion but any letter will do. (In principle, certain non-alphanumeric characters can also be used but that seems like a bad idea to me.) The variable letter is preceded with a single percent sign when using the command line directly or double percent signs in a batch file. Thus the statement in a batch file looks like this:

```
for %%X in (set) do (command)
```

What makes the "For" statement so powerful is the variety of objects that can be put in the set of things that the command iterates through, the availability of wildcards, and the capability for parsing files and command output. A number of switches or modifiers are available to help define the type of items in the set. Table I lists the switches. They are listed in upper case for clarity but are not case-sensitive.

Table I. Modifying switches used with FOR

Switch	Function
/D	Indicates that the set contains directories.
/R	Causes the command to be executed recursively through the sub-directories of an indicated parent directory
/L	Loops through a command using starting, stepping, and ending parameters indicated in the set.
/F	Parses files or command output in a variety of ways

I will consider a number of examples that illustrate the use of "For" and its switches.

### Simple iteration through a list

The set of things that are to be used can be listed explicitly. For example, the set could be a list of files:

```
for %%X in (file1 file2 file3) do command
```

(Care must be taken to use correct paths when doing file operations.) A different example where the set items are strings is:

```
For %%X in (eenie meenie miney moe) do (echo %%X)
```

Wildcards can be also be used to denote a file set. For example:



[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

```
for %%X in (*.jpg) do command
```

This will carry out the command on all files in the working directory with extension "jpg". This process can be carried further by using several members in the set. For example to carry out a command on more than one file type use:

```
for %%X in (*.jpg *.gif *.png *.bmp) do command
```

As always, keep in mind that the command line may choke on file names with spaces unless the name is enclosed correctly in quotes. Therefore, you might want to use "%X" in the "command" section.

## Looping through a series of values

The well known action of stepping through a series of values [that was discussed](#) in connection with "if" and "Goto" statements is succinctly done with the switch /l (This switch is an "ell", not a "one") . The statement has the form:

```
for /l %%X in (start, step, end) do command
```

The set consists of integers defining the initial value of X, the amount to increment (or decrement) X in each step, and the final value for X when the process will stop. On the [previous page](#), I gave an example batch file that listed all the numbers from 1 to 99. If we use a "For" statement, that task can be accomplished with one line:

```
for /l %%X in (1,1,99) do (echo %%X >> E:\numbers.txt)
```

The numbers in the set mean that the initial value of X is 1, X is then increased by 1 in each iteration, and the final value of X is 99.

## Working with directories

If you wish to use directories in the variable set, use the switch /d. The form of the command is

```
for /d %%X in (directorySet) do command
```

An example that would list all the directories (but not sub-directories) on the C: drive is

```
for /d %%X in (C:\*) do echo %%X
```

## Recurring through sub-directories

If you want a command to apply to the sub-directories as well as a parent directory, use the switch /r. Then the command has the form:

```
for /r [parent directory] %%X in (set) do command
```

Note that you can designate the top directory in the tree that you want to work with. This gets around the often cumbersome problem of taking into account which is the working directory for the command shell. For example the statement:

```
for /r C:\pictures %%X in (*.jpg) do (echo %%X >> E:\listjpg.txt)
```

will list all the jpg files in the directory C:\pictures and its sub-directories. Of course, a "dir" command can do the same thing but this example illustrates this particular command.

## Parsing text files, strings, and command output

Now we come to a truly powerful switch that was not even dreamed of back in the DOS days of yore. The switch /f takes us into advanced territory so I can only indicate the many aspects of its application. Things become rather complex so those who are interested should consult programming books or the [Microsoft documentation](#). However, here is a brief sketch of what's involved.

This version of the "For" command allows you to examine and parse text from files, strings, and command output. It has the form

```
for /f [options] %%X in (source) do command
```

"Options" are the text matching criteria and "source" is where the text is to be found. One of the interesting applications is to analyze the output of a command or commands and to take further action based on what the initial output was.

[Back to top](#)

# The Command Line in Windows

## Site navigation

powered by [FreeFind](#)

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the](#)

## Introduction to the Command Prompt

### DOS is dead, long live the command line

Before proceeding further, I wish to clear up a common misconception that the command prompt in Windows XP is the same as DOS. Even some experts who know better sometimes refer to the command prompt as a "DOS window". In fact, Microsoft itself isn't always careful about the distinction. There are some superficial resemblances and some commands with the same name but in fact the old 16-bit DOS is dead. All remnants of DOS are totally gone from the Windows XP kernel (there is a DOS *emulator* for legacy programs). Windows XP is a 32-bit protected memory system with a totally different approach from the DOS/9X/Me family. The command line in XP has many more capabilities and none of the 16-bit limitations like the restriction to the DOS 8.3 file name format. The augmented capabilities make the command line a powerful tool.

### The command prompt window

The command prompt is run from its own window by invoking the Windows XP command interpreter that is provided by the file *cmd.exe* located in the folder `\Windows\System32\`. (The old DOS command interpreter is *command.com*.) If you look in this folder you may also see several files that look suspiciously like some of the old DOS files. They are, however, different 32-bit versions with many new features. The command prompt window can be opened by entering "cmd" (without quotes) into Start-Run or through Start-All Programs-Accessories. A black and white window (the [colors can be changed](#)) containing the command prompt will open. The window looks just like the old DOS window but don't be fooled, it isn't. Note that it is possible to open several windows containing command prompts, all running independently. It is even possible to run a separate command prompt shell inside another command prompt window.

### Internal and external commands

There are two kinds of commands that can be run from the command prompt. There are the internal commands that are built into the command interpreter like "del" and "dir". These commands can only be run from a command prompt (or by invoking the command interpreter in some other way). They are listed in the table below. There is also a large list of external commands that use an additional executable file that can be run from either the command prompt or the [Start-Run line](#). Details of the various commands are available in several places. In the Professional version of Windows XP there is a help file *ntcmds.chm*, which has details of all the commands and their many switches. The help file can be opened by entering (without the quotes) "hh ntcmds.chm" into Start-Run. It may or may not be in the Home Edition, depending on what setup you have. However, in both versions a list of many (but not all) of the commands available can be obtained by entering "help" (without quotes) into a command prompt. For more detail on a specific command, enter "help command-name" or "command-name /?". For example to get information on the command *xcopy*, enter "help xcopy" or "xcopy /?". Microsoft keeps moving things, but the last time I checked they had a [command line reference at this link](#). Some of the commonly used commands are discussed on [this page](#) and in the [list given here](#).

Table 1. Internal commands in the command shell

assoc	dir	move	set
-------	-----	------	-----

[command shell](#)[Tskill and Taskkill](#)[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

break	echo	path	setlocal
call	endlocal	pause	shift
cd	exit	popd	start
cls	for	prompt	time
color	ftype	pushd	title
copy	goto	rd	type
date	if	rem	verify
del	md	ren	volume

## Some symbols that are used

In addition to the commands, there are several symbols that are used. These modify or combine the actions of commands. The table below gives a list.

*Table II. Symbols used with commands*

Symbol	Function	Example
>	Sends output to a named file. If file does not exist, it creates one. Overwrites existing file	command > somefile
>>	Appends output to contents of a named file or creates a file if none exists	command >> somefile
<	Uses contents of a named file as input to a command	command < somefile
	Sends ("pipes") the output of command1 to the input of command2	command1   command2
&	Used to combine two commands. Executes command1 and then command2	command1 & command2
&&	A conditional combination. Executes command2 <i>if</i> command1 completes successfully	command1 && command2
::	Command2 executes only if command1 does not complete successfully.	command1 :: command2
@	Used in batch files at the beginning of a line to turn off the display of commands	@echo off

The most commonly used symbols are the two redirection symbols ">" and ">>" and the so-called pipe, "|". (Just to make sure there is no confusion, the "pipe" is the symbol above the back slash on most keyboards. On keyboards it has a break in the middle but the break does not always show when you type the symbol. A special code is used to show it on a Web page.)

A frequent use of the redirection is to save some output to a text file. For example the command

```
dir somefolder > somefile.txt
```

sends a list of the files in "somefolder" to a text file "somefile.txt". More about this type of use is [on this page](#). A common use of the "pipe" is to control the screen display of some command with a lot of output. For example, if you want to check the list of files in a folder with many files, you can display one full screen at a time by piping to the command "more"

```
dir somefolder | more
```

[Back to top](#)

# The Command Line in Windows

## Command Line List and Reference

The complete list of possible commands is quite large. Collected here is a selection of those that I believe are likely to be the most applicable to home PC use.

Many of the commands listed below are also discussed in more detail elsewhere on this site as is indicated by the link "details here". Most commands have switches that are not given in the table. For more information, open a command prompt and enter "*commandname /?*" (without quotes). Detailed information about these commands and a larger list is available at this [Microsoft reference](#).

Selected list of commands

Command	Description	Example
assoc	Displays or modifies file name extension associations. Used alone, displays a list of all the current file name associations	<a href="#">details here</a>
at	Schedules commands and programs to run on a computer at a specified time and date. Requires the Schedule service. Superseded by <i>schtasks</i>	
attrib	Configures file attributes <i>read only, hidden, system</i>	<a href="#">details here</a>
bootcfg	Used to repair or edit the <i>boot.ini</i> file	<a href="#">details here</a>
cd or chdir	Displays the name of the current directory or changes the current folder	cd folderpath
chkdsk	Checks hard drives for errors. With switches, does repairs.	<a href="#">details here</a>
cls	Clears the screen	cls
copy	Copies a file from one location to another	copy somefile somefolder
del	Deletes one or more files	<a href="#">details here</a>
dir	Displays a list of a folder's files and subfolders	<a href="#">details here</a>
echo	Used to display a message or to turn off/on messages in batch scripts	echo message
exit	Exits batch script or current command control	exit
fc	Compares two files and displays the differences between them	fc file1 file2
for	Runs a specified command for each item in a set	<a href="#">details here</a>
fsutil	Displays and configures certain file system properties. A suite of various commands	<a href="#">details here</a>
ftype	Displays or modifies file types used in file name extension associations	<a href="#">details here</a>
getmac	Returns the media access control (MAC) address for your network card	getmac

### Site navigation

powered by [FreeFind](#)

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and  
reference](#)

[Commands that everybody  
can use](#)

[Configuring the command  
prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command  
line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

goto	Directs the Windows command interpreter to a labeled line in a batch program	<a href="#">details here</a>
if	Performs conditional processing in batch programs	<a href="#">details here</a>
ipconfig	Displays all current TCP/IP network configuration values and refreshes Dynamic Host Configuration Protocol (DHCP) and Domain Name System (DNS) settings	<a href="#">details here</a>
md or mkdir	Creates a directory (folder) or subdirectory (subfolder)	<a href="#">details here</a>
more	Displays one screen of output at a time. Used with another command that has extensive output	<i>command</i>  more
move	Moves a file from one folder to another	<a href="#">details here</a>
net	A suite of various networking and service commands	<a href="#">details here</a>
netsh	Network services shell . Complex suite of commands.	<a href="#">details here</a>
netstat	Displays active TCP connections, ports on which the computer is listening, Ethernet statistics, the IP routing table, statistics for the IP, ICMP, TCP, and UDP protocols	<a href="#">details here</a>
path	Sets the command path in the PATH environment variable, which is the set of directories used to search for executable files	<a href="#">details here</a>
pathping	Provides information about network performance and conditions at intermediate hops between a source and destination	<a href="#">details here</a>
pause	Used in batch scripts	pause
ping	Checks connectivity to other networked computers, routers, or Internet sites	<a href="#">details here</a>
popd, pushd	Changes the directory being referenced in a command prompt. Pushd changes the directory and stores the previous directory. Popd changes the current directory to the directory stored by the pushd command	pushd somefolder popd
powercfg	Manages the power settings such as hibernation. Has numerous switches	
reg	Adds, changes, and displays registry entries. A suite of various commands	<a href="#">details here</a>
rd or rmdir	Deletes a directory (folder)	<a href="#">details here</a>
ren or rename	Changes the name of a file or a set of files	<a href="#">details here</a>
sc	Used to obtain information about services and to configure them. A suite of various commands	<a href="#">details here</a>
schtasks	Schedules commands and programs to run periodically or at a specific time	
set	Displays, sets, or removes environment variables	<a href="#">details here</a> and also <a href="#">here</a>
sfc	System file checker scans and verifies the versions of all protected system files	sfc /scannow
shutdown	Shuts down or restarts a computer	<a href="#">details here</a>
start	Starts an application or opens a new command window	<a href="#">details here</a>
subst	Associates a folder with a drive letter	<a href="#">details here</a>



systeminfo	Displays detailed configuration information about a computer and its operating system	systeminfo   more
taskkill	Ends one or more tasks or processes	<a href="#">details here</a>
tasklist	Displays a list of applications and services with their Process ID (PID) for all tasks running	<a href="#">details here</a>
tree	Graphically displays the directory structure of a folder or drive	tree somefolder
type	Displays the contents of a text file	type somefile.txt
xcopy	Powerful command with many switches for copying and backing up files and folders	<a href="#">details here</a>

[Back to top](#)

# The Command Line in Windows

## Site navigation

powered by [FreeFind](#)

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

## Commands that everybody can use

Since the command prompt is often used by systems administrators, many of the commands are specialized for networks and administration. However, the average PC user is not left out; there are some powerful commands for everyday tasks that anybody will find useful. Here we discuss some examples.

For convenience in the examples below, I will use simple names for files and folders without indicating the full path. Please remember that paths for all commands are relative to the folder indicated in the command prompt itself, unless explicitly indicated otherwise. The default reference folder is usually C:\Documents and Settings\(\log on name)\. Thus if a reference is made to a file "somefile.txt", the computer assumes you mean C:\Documents and Settings\(\log on name)\somefile.txt. If what you really want is C:\someotherfolder\somefile.txt, then you must enter the entire name with the correct path. Note that paths containing spaces have to be enclosed in quotes. A convenient way to be able to open the command prompt in a folder of your choice can be obtained by installing the Microsoft PowerToy "Command Prompt Here". The right-click context menu will then contain an entry for opening a command prompt in any selected folder. The accessory can be [downloaded here](#). Another way to get file names into the command line without having to type a long path is by using drag and drop. Open a command window and enter the command you want with a space after it. Then use Windows Explorer to open the folder containing the file you want to use. Drag the file over to the command window and drop it. (Drag and drop does not work in Vista.)

### Power deleting with the expanded "Del" command

The delete command "del" now has a switch "/s" that provides for deletions in subfolders. Thus the command

```
del /s myfolder\*
```

will delete all files in *myfolder* and all files in any subfolders of *myfolder*. Note the convenient asterisk wildcard "\*", which allows for multiple deletions in a single user operation. Together with the switch "/s", a single "del" command can clean out Temp folders and do other useful housekeeping chores. Another switch "/f" will force the deletion of read-only files. If you are sure about what is being deleted, add the "/q" switch to run in quiet mode so you aren't asked if you really want to delete for every single file. *Because of its power, this extended del command has to be used carefully. Any files deleted this way do not go into the Recycle Bin but are permanently removed.*

To delete only files with a particular extension EXT, use the command

```
del /s myfolder\*.ext
```

Note that this command will delete all files of a particular type in a folder and all its subfolders. One example of its use is to get rid of all temporary files with the extension TMP.

### Enhanced management of folders (directories)

The command "rmdir" or its twin "rd" are also expanded compared to the old DOS version. To remove folders (directories), use "rmdir" (just "rd" will also work). Folders must be empty before they can be deleted. However, there is a switch "/s" for deleting subfolders and with this switch files are also deleted. It is like the old command "deltree". The command

[Tips for using the](#)[command shell](#)[Tskill and Taskkill](#)[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

```
rd /s myfolder
```

will remove "myfolder" and everything in it. Adding the switch /q for a "quiet" mode is also possible.

Options have also been added to the command for making folders. The command "mkdir" or "md" can now make a whole tree.

For example, consider the command

```
md new\new1\new2
```

This will create a folder "new" with a subfolder "new1" that in turn has a subfolder "new2".

## Xcopy- Powerful and versatile way to copy or back up files

For copying large numbers of files or backing up a large folder, the command line is often the fastest and easiest way to go. The command "xcopy" is a very useful and powerful command for this purpose. (The old DOS also had an "xcopy" command but it sometimes had trouble with long file names. The XP version has no such problems.) "xcopy" comes with an alphabet soup of [assorted switches](#) that give it great versatility for use as a file backup utility. Enter "xcopy /?" in a command prompt to see them all. For example with the switch "/d:[mm-dd-yyy]" only files changed after a given date are copied. A command that will copy all files from the folder *myfolder* that have changed since June 1, 2003 to the folder *mybackup* is given by

```
xcopy myfolder mybackup /d:06-01-2003
```

If no date is specified the switch "/d" will copy all files that have changed at any time. As is true in general, if there are spaces in a name, the path and file name have to be enclosed in quotes. Adding the switch "/s" provides that subfolders and their contents will also be copied. Other switches provide for read-only and hidden files. Altogether, there are something like 27 command line options for *xcopy*. More information is [on this page](#).

## How to list files with "dir"

The "dir" command is not new but it remains one of the most useful for average PC users. If you have ever wanted to make a list of all the files contained in a particular folder, you will have discovered that the Windows GUI provides no easy way to do this. It is very straightforward, however, when using the command line. The command "dir myfolder" will list the files and folders contained in *myfolder*. Again, there are switches that provide for various modifications of the command. For example "/h" will show hidden files and "/s" will list the contents of sub-folders in addition to those of the main folder. Of course, the normal output of the command is to the screen. To output to a file instead of the screen, the command is

```
dir myfolder >listmyfolder.txt
```

where ">" is the redirection symbol. The file "listmyfolder.txt" will be created by the command. Output can also be redirected straight to a printer on the LPT1 port but I think it is better to first create a file and then print from there if hardcopy is desired. These lists can be a lot longer than you might think. (These days most printers are on USB, anyway.) This use of *dir* is also available in Windows 98/Me.

The *dir* command can also be used to list only files with a given extension. For example

```
dir myfolder\*.doc /s >listmyfolder.txt
```

will list all Microsoft Word files in *myfolder* and its subfolders. This form of the command will also list all the directories and sub-directories. If you want a list of filenames only, add the switch /b. The filename will include the full path.

## Renaming files with "ren"

Renaming a large number of files can be tedious. The command *ren* (also written *rename*) is somewhat limited but its ability to use wild cards can sometimes be useful. The basic command is

```
ren file1 file2
```

The renamed file has to stay in the same folder as the original; this command cannot move files to another folder. The wildcard

capability can be used to change the extension of all files of a certain type. Thus

```
ren *.txt *.doc
```

will rename all text files to have a DOC extension.

## Moving files

The command "move" takes a file from one folder and puts it in another. Its ability to use wild cards makes it useful for moving all the files of a given type to another location. A command of the type

```
move /y folder1\*.mp3 folder2\
```

will move all MP3 files from *folder1* to *folder2*. The switch "/y" is used if you want to prevent the system from asking if it should overwrite existing files of the same name. To prevent overwriting, use the switch "/-y".

## Changing file attributes with "attrib"

In addition to their actual content, files also have a set of properties that characterize them (sometimes called [metadata](#).) One set of these properties are the four attributes *read-only*, *hidden*, *system*, and *archive*. The archive attribute is primarily used in backup procedures but the other three can be encountered in various contexts. These attributes can be turned on or off with the command "attrib" and its switches. The great utility of this command is that it can act on subfolders with its switch "/s" and can use the wildcard "\*\*". For example, to clear the read-only, system, and hidden attributes from a file use

```
attrib -r -s -h somefile
```

This operation is not uncommon when system files have to be edited. To restore the attributes the command is

```
attrib +r +s +h somefile
```

A common situation where it is desirable to clear the read-only attribute from many files is when files are copied from a CD. By default CD files are normally marked read-only. This can interfere with editing. To clear the read-only attribute from all files in a folder and its subfolders use

```
attrib -r somefolder\*. * /s
```

If you want to process all files of a certain type such as MP3, use

```
attrib -r somefolder\*.mp3 /s
```

## How to make your favorite folder easily accessible in Windows dialog boxes

Another older command that I find handy is "subst". There are certain folders that I use over and over and I like to have ready access to them. One way to do this is to use the command "subst" to assign a drive letter to a folder. Since drives are at the top of My Computer and any folder trees in browse lists, it makes the folder very easy to get to. To map the "Z:" drive to a folder "myfolder", enter

```
subst z: myfolder
```

Unless you are working from the parent folder of "myfolder" you will need the full path for it. The assignment only lasts until the user logs off or the computer is shut down so I have a one-line batch file in my Startup folder that reinstates the assignment.

## Other commands

There are so many commands that we can only give the briefest sketches here. A list of some that I think might be useful on home computers is [given here](#). I urge you to look into as many as possible since I feel you will be surprised at how useful some can be. For example, there are a whole host of commands for checking network functions and for use on the Internet. (Many have names that begin with "net...") There are new ones as well as versions of the well-known commands such as "ping" and "tracert". As more and more people have home networks, these commands are assuming greater significance to the average

PC user. Go to [this page](#) for more discussion of some networking commands.

## Useful command line programs in scripts

Average PC users are not always comfortable using the command line and I have used VBScripts to wrap some of the examples discussed above with an easy-to-use graphical interface. Description of the free scripts and instructions for their use and downloading are [at a sister site](#).

[Back to top](#)

# The Command Line in Windows

## Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the command shell](#)

[Tskill and Taskkill](#)

## Configuring the Command Prompt Window

The interface to the command shell can be customized in a number of ways and these are discussed.

The default settings for the window that displays a command prompt can be changed to suit individual preferences. Among the properties that can be changed are those that affect the appearance such as window size, background and foreground color, and font style. Others that affect operation such as the editing mode and command history buffer can also be configured.

### The command prompt properties dialog

Changing the settings for the command prompt can be done through the properties dialog box. This can be reached in the menu that is opened by left-clicking the icon in the left-top corner or by right-clicking anywhere in the title bar at the top of a command prompt window. In the menu that opens choose "Properties". Figures 1 and 2 show the procedure.

Fig. 1. Menu for command window

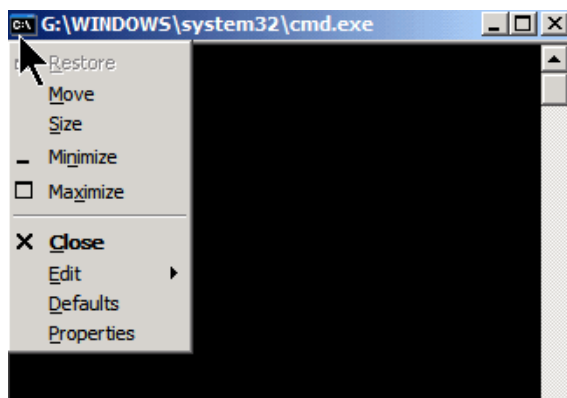
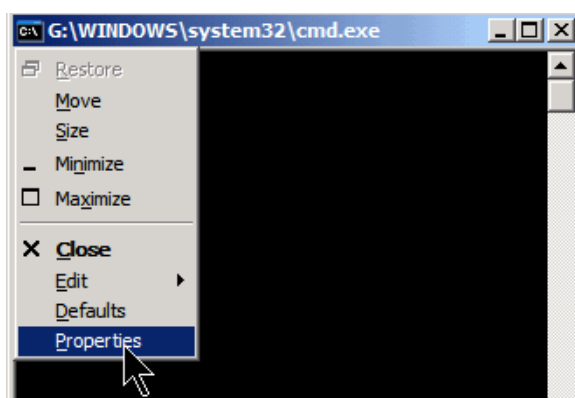


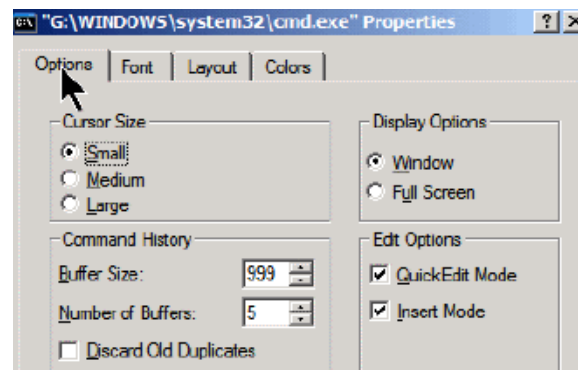
Fig. 2. Entry for properties dialog



The properties dialog box contains four tabs. Each of these is discussed below.

### Options tab

The figure on the right shows the dialog box for the Options tab. Settings here include provision for configuring cursor size and window or full-screen display. The "Quick Edit" mode allows you to use the mouse to cut and paste text to and from the command window. You can also choose insert or overwriting for editing by checking or unchecking "Insert Mode".



### Command History buffer

A new feature is the "Command History" buffer. This buffer stores the previous commands that you have entered so that you do



[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

not have to retype them (reminiscent of Doskey). The default value is 50 but can be made as large as 999 (remember that they do take RAM). The up-and-down arrow keys can be used to navigate the previously entered commands. Alternatively, pressing the F7 key will display a pop-up window with a list of the commands.

## Font tab

This tab is more or less self-explanatory. On most machines, there is a choice of two font faces, Raster (the default) and Lucinda Console. Lucinda Console is more versatile. A selection of font sizes is available. As is true in general about settings, changes in font can be made for the present window only or for all command windows with the same title.

## Layout tab

The window size and position can be set here. There are also settings for the screen buffer size. This determines how many lines back that you can scroll. (Incidentally, this ability to scroll back to previous commands was not present in DOS. Once something was off the screen, you couldn't scroll back.)

## Color tab

The dialog box for configuring colors is shown on the right. The default colors for a command window and its fonts are the old black and white combination. However, other color combinations are available. The colors for pop-up windows associated with the command window can also be configured. As shown in the figure on the right, colors can be changed by clicking on the appropriate choice or by entering numerical values in the 256 color RGB notation. In the figure, I have chosen a blue screen background with yellow screen text as illustration. Color changes can be made for the current screen only or applied to all command windows.

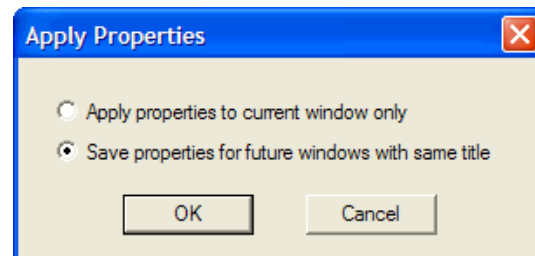
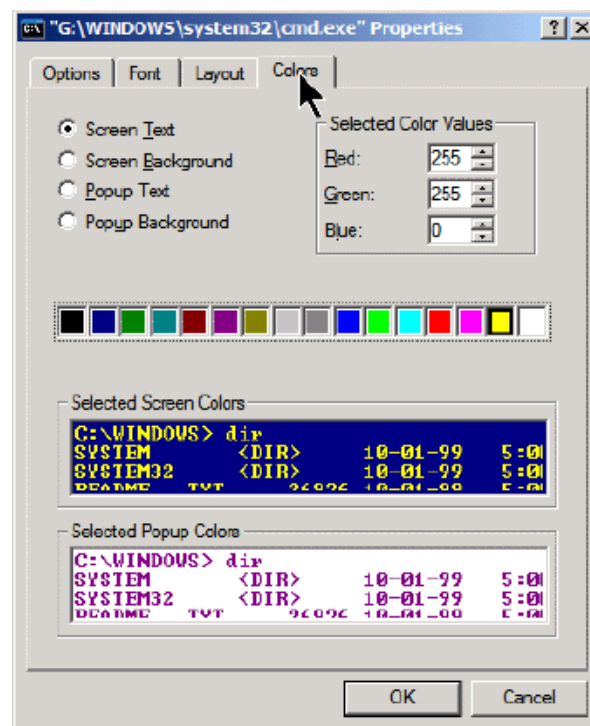
Screen colors can also be changed with the "color" command or a command-line switch directly modifying the command interpreter. Both these methods are discussed below.

It is also possible to have shortcuts that open the command line with more than one color scheme and this is discussed in the shortcut section below.

## Applying changes to all command windows

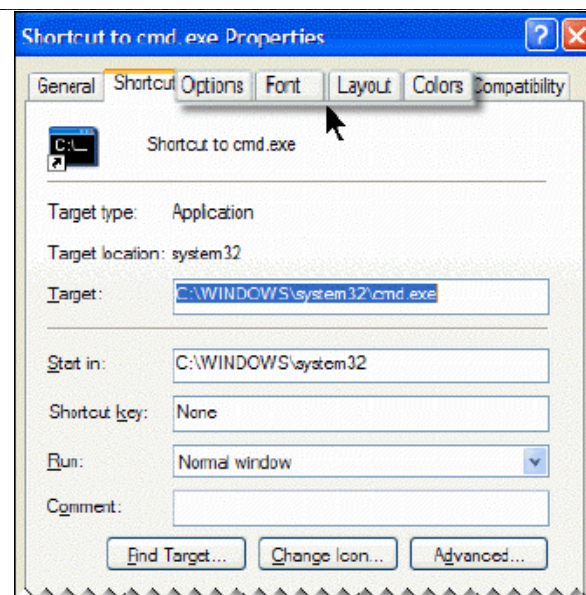
Note that changes apply to the current command window only unless you specify otherwise. When making configuration changes, you will be shown the dialog box on the right. To make changes stick after the current command session is closed, check the button by the entry, "Save properties for future windows with same title".

## Configuring shortcuts for the

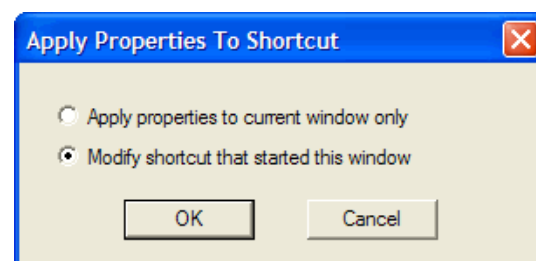


## command prompt

If you use the command prompt a great deal, you may want to place a shortcut on the desktop or in the Quick Launch bar. After creating the shortcut, you can configure the properties of the command window that it will open in much the same way as discussed above. Right-click the shortcut and choose "Properties" from the context menu. Four of the the tabs at the top of the properties sheet will be the same as those discussed above. They are marked in the figure on the right. You can apply whatever customization you like and it will apply only to command windows that are opened from this particular shortcut.



Another way to configure the properties of a particular shortcut is to apply the changes in a command window that has been opened with that shortcut. After configuring the changes to the open command window by the methods discussed in sections above, the dialog box shown on the right will open. Check the button by the entry, "Modify shortcut that started this window".



## The "color" command

The screen colors can be configured from the command line itself with the command "color" followed by a two-digit hexadecimal number. The first digit determines the background and the second determines the text color. The table below shows the relationship between the hex numbers and colors.

Table 1. Hexadecimal color codes

0 = Black	8 = Gray
1 = Blue	9 = Light Blue
2 = Green	A = Light Green
3 = Aqua	B = Light Aqua
4 = Red	C = Light Red
5 = Purple	D = Light Purple
6 = Yellow	E = Light Yellow
7 = White	F = Bright White

For example the command "color 1E" will give a blue background with yellow text. Settings made this way apply only to the current session. Entering "color" with no argument will return the system to the starting colors.

## Switches for the command interpreter *cmd.exe*

Another way to modify the behavior of the command prompt is to use switches with the command interpreter *cmd.exe*. The syntax for *cmd.exe* is:

```
cmd [[{/c|/k}] [/s] [/q] [/d] [{/a|/u}] [/t:fg] [/e:{on|off}] [/f:{on|off}] [/v:{on|off}]
```

*string*]

Table II describes the parameters briefly. See the Windows Help and Support Center for more detail.

Table II. Switches for *cmd.exe*

Parameter	Description
<i>/c</i>	Carries out the command specified by <i>string</i> and then exits
<i>/k</i>	Carries out the command specified by <i>string</i> and stays open
<i>/s</i>	Modifies the treatment of <i>string</i> after <i>/c</i> or <i>/k</i> . See Windows Help for details
<i>/q</i>	Turns the echo off. Default is echo on
<i>/d</i>	Disables execution of AutoRun commands
<i>/a</i>	Creates ANSI output (the default)
<i>/u</i>	Creates Unicode output
<i>/t:f g</i>	Sets the foreground <i>f</i> and background <i>g</i> colors. The hex codes for <i>f</i> and <i>g</i> are in Table I.
<i>/e:on /e:off</i>	Enables or disables commands extensions. The default is "on"
<i>/f:on /f:off</i>	Enables or disables file and directory name completion
<i>/v:on /v:off</i>	Enables or disables delayed environment variable expansion
<i>string</i>	Specifies the command you want to carry out

[Back to top](#)

# The Command Line in Windows

## Doskey in the Windows XP Command Shell

### Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the](#)

[command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)

[Vista command list](#)

The Windows XP command console contains an accessory named after the old command-line utility called Doskey. Its functions are discussed.

As the name implies, Doskey was part of the old DOS command shell. It provided command-line editing, history functions for recalling commands, and a way to write macros. A similarly-named tool is still present in the 32-bit command shell provided by *cmd.exe* but its functions have been largely superseded. For example, if extensions are enabled (the default setting), a [buffer that stores command-line entries](#) provides for the easy recall of up to 50 previously entered commands without need for Doskey. Using the Up and Down arrow keys will navigate among previously entered commands. Unless you are working for long periods of time within the command console, I also see little advantage of macros over batch files. I include this page for completeness but I have not personally used Doskey for years.

### Syntax and functions

The figure below summarizes the syntax for Doskey and its switches and functions.

```
C:\>doskey /?
Edits command lines, recalls Windows XP commands, and creates macros.

DOSKEY [/REINSTALL] [/LISTSIZE=size] [/MACROS[:ALL ; :exename]]
        [/HISTORY] [/INSERT ; /OVERSTRIKE] [/EXENAME=exename] [/MACROFILE=filename]
        [macroname=[text]]

  /REINSTALL           Installs a new copy of Doskey.
  /LISTSIZE=size      Sets size of command history buffer.
  /MACROS              Displays all Doskey macros.
  /MACROS:ALL         Displays all Doskey macros for all executables which have
                    Doskey macros.
  /MACROS:exename     Displays all Doskey macros for the given executable.
  /HISTORY            Displays all commands stored in memory.
  /INSERT             Specifies that new text you type is inserted in old text.
  /OVERSTRIKE         Specifies that new text overwrites old text.
  /EXENAME=exename   Specifies the executable.
  /MACROFILE=filename Specifies a file of macros to install.
  macroname          Specifies a name for a macro you create.
  text               Specifies commands you want to record.

UP and DOWN ARROWS recall commands; ESC clears command line; F7 displays
command history; ALT+F7 clears command history; F8 searches command
history; F9 selects a command by number; ALT+F10 clears macro definitions.

The following are some special codes in Doskey macro definitions:
$!   Command separator. Allows multiple commands in a macro.
$1-$9 Batch parameters. Equivalent to %1-%9 in batch programs.
$*   Symbol replaced by everything following macro name on command line.
```

### More information

Further description of Doskey can be found at [this Microsoft reference](#).

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

< [Home](#) | ©2002-2010 Victor Laurie

# The Command Line in Windows

## Site navigation

[Home](#)
[Assoc and Ftype](#)
[Batch files- basics](#)
[Batch files - branching](#)
[Batch files- iterating](#)
[Command Line-](#)
[Introduction](#)
[Command line list and reference](#)
[Commands that everybody can use](#)
[Configuring the command prompt window](#)
[Doskey](#)
[File system utility- Fsutil](#)
[Net Services](#)
[Netstat](#)
[Network Services Shell](#)
[PowerShell](#)
[Recovery Console](#)
[Recovery Console- Commands](#)
[Registry editor console](#)
[Scripts in the command line](#)
[Server 2003 tools for XP](#)
[Service Controller](#)
[Command \(SC\)](#)
[Shell command](#)
[Start-Run line](#)
[Support tools](#)
[Tasklist](#)
[TCP/IP networking tools](#)
[Tips for using the command shell](#)
[Tskill and Taskkill](#)

## File System Utility- Fsutil.exe

The file system utility "fsutil" is a suite of command-line operations for displaying and managing certain file and drive properties. Some applications are described.

*Fsutil* is an advanced tool intended primarily for system administrators but more experienced PC users will also find that it has a number of possible applications. Some that I think may be of interest are discussed. The tool is present in both Windows XP and Vista and is primarily of use for NTFS systems. It requires administrative privileges.

## Subcommands

*Fsutil* contains a suite of subcommands, which are listed in Table I. Each of these subcommands may in turn have additional subcommands of its own. Many of these are quite specialized but some that are of more general interest are discussed in sections further on. Windows Vista has two additional subcommands not present in Windows XP and these are indicated in the table. They are included for completeness but are of limited interest to most PC users.

Table I. Subcommands for Fsutil

Subcommand	Description
behavior	Manages the settings for generating 8.3 character-length file names and for, updating the last access timestamp. Manages the amount of disk space reserved for the Master File Table.
dirty	Queries or sets a volume's dirty bit.
file	Finds a file by its security identifier, queries allocated ranges for a file, sets a file's short name, sets a file's valid data length, or sets zero data for a file.
fsinfo	Lists all drives, queries the drive type, queries volume information, queries NTFS-specific volume information, or queries file system statistics.
hardlink	Creates a hard link
objectid	Manages object identifiers
quota	Manages disk quotas on NTFS volumes
repair (Vista)	Self healing management
reparsepoint	Queries or deletes reparse points
sparse	Manages sparse files
transaction (Vista)	Transaction management
usn	Manages the update sequence number (USN) change journal
volume	Dismounts a volume or queries to see how much free space is available on a disk.



[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

## Examples of applications of Fsutil

Here is a selection of applications of *Fsutil* that I have seen mentioned most often on the Web. If I have omitted one of your favorites, let me know and I will add it to the list.

### Applications of the subcommand "behavior"

The subcommand "behavior" may be the one that an ordinary PC user is most likely to encounter. There are several features of the NTFS system that can be configured to possibly speed up the system and tweaks involving this subcommand are fairly commonly mentioned on the Web and in books. The subcommand "behavior" has two further subcommands. They are "query", which displays the current values for option settings and "set", which is used to change settings. The options that are available are shown in Table II. Options that are available only in Vista are indicated. Commands have the form

```
fsutil behavior query {option} {value}
```

and

```
fsutil behavior set {option} {value}
```

Commands involving "set" can change Registry entries and may require a reboot to take effect.

*Table II. Options for "fsutil behavior"*

Option	Possible Values
allowextchar	0 or 1
disable8dot3	0 or 1
disablecompression (Vista)	0 or 1
disablelastaccess	0 or 1
disableencryption (Vista)	0 or 1
encryptpagingfile (Vista)	0 or 1
mftzone	0 through 4
memoryusage (Vista)	0, 1, 2
quotanotify	1 through 4294967295 seconds
SymlinkEvaluation (Vista)	Various

### Find the current settings

To display the current settings on a system, use a command with "query" such as:

```
fsutil behavior query disablelastaccess
```

### Disable short file names to speed up Windows

One tweak that is mentioned fairly often disables the creation of short 8.3 format file names. By default both Windows XP (NTFS) and Vista create file names in the old 8.3 format for all files in addition to whatever the regular name is. This is to ensure compatibility with some programs that still linger and require the old file naming format from DOS days. If you are sure that you do not have any 16-bit programs or programs that require the old format, disabling this extra name creation can possibly speed up your system. The command is:

```
fsutil behavior set disable8dot3 1
```

Note that the Windows XP [environment variables](#) %TEMP% and %TMP% typically use short names. Program installations sometimes use these variables. If this change causes trouble and you need to restore the short name function, the command is

```
fsutil behavior set disable8dot3 0
```

### Disable timestamp for last access to a file to speed up Windows

Another frequently seen recommendation is to disable the setting that keeps track of the last time a file was accessed.

Removing the necessity for the system to keep reading and writing this information may speed up Windows Explorer. The command is:

```
fsutil behavior set disablelastaccess 1
```

Note that some backup programs may need this information. If you wish to restore the timestamp, the command is:

```
fsutil behavior set disablelastaccess 0
```

### Disable Encrypting File System in Windows Vista

Windows Vista Business and Ultimate come with a feature called the [Encrypting File System](#) (EFS). This can be fairly demanding of resources and if you have a marginal system that does not require this security feature, you can turn EFS off. The command is

```
fsutil behavior set disableencryption 1
```

To restore EFS, use the command

```
fsutil behavior set disableencryption 0
```

(Although Windows XP Professional has EFS, this particular command is not available.)

### Change the size of the Master File Table

NTFS uses a Master File Table (MFT) to store information about folders and files. Entire small files may even be included. By default, 12.5% of the volume is allocated to the MFT. Unless you have a small volume or a very large number of files, the default allocation should suffice. However, if you need more space for the MFT, you can use the command

```
fsutil behavior set mftzone n
```

The parameter "n" can be 0 to 4. Table III shows the MFT allocation corresponding to each value of "n".

*Table III. MFT space allocations*

value for n	Space allocation for MFT
0	No setting. Uses the default of 12.5%
1	12.5%
2	25%
3	37.5%
4	50%

## Displaying some drive properties

*Fsutil* has some very advanced methods for managing hard drives but I will mention only a few of more general interest.

### Obtaining drive lists for a computer

There are other ways of doing this but you can obtain a list drives by letter type by using the command:

```
fsutil fsinfo drives
```

### Obtaining drive type

You can the drive type for a given drive with the command:

```
fsutil fsinfo drivetype D:
```

The command will return "Fixed drive", "CD-ROM drive", or "Removable drive".

### Obtaining some general volume information

You can obtain a list of various volume parameters with:

```
fsutil fsinfo volumeinfo C:
```

### Determining amount of free space on a drive

To determine the amount of free space on a drive, use the command

```
fsutil volume diskfree C:
```

[Back to top](#)

# The Command Line in Windows

## Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console- Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the command shell](#)

[Tskill and Taskkill](#)

## Net Services Commands in Windows XP

A large family of commands beginning with the initial string "net" are available in Windows XP Professional. They are listed here.

The Net Services suite of commands is described in the *Windows Help and Support Center*. Enter "net services" to obtain a list of the services and their syntax for usage. You can also see a list of all available net commands by typing "net /?" at a command prompt. Syntax help is obtained by typing "net help {command}". For example, for help with the "net stop" command, type

```
net help stop
```

The list of commands and a brief description of each is given below. Some of these commands duplicate functions available in other ways, such as in the [Netsh suite](#) or the [Service Controller](#).

- *Net accounts* - Updates the user accounts database and modifies password and logon requirements for all accounts.
- *Net computer* - Adds or deletes computers from a domain database
- *Net config* - displays a list of configurable services
- *Net continue* - Continues a service that has been suspended by *net pause*
- *Net file* - Displays the names of all open shared files on a server
- *Net group* - Adds, displays, or modifies global groups in domains
- *Net help* - Provides a list of network commands and topics for which you can get help
- *Net helpmsg* - Explains why an error occurred and provides problem-solving information
- *Net localgroup* - Adds, displays, or modifies local groups
- *Net name* - Adds or deletes a messaging name
- *Net pause* - Pauses services that are currently running.
- *Net print* - Displays information about a specified print queue, displays information about all print queues hosted by a specified print server, displays information about a specified print job, or controls a specified print job.
- *Net send* - Sends a messenger service message
- *Net session* - Lists or disconnects sessions
- *Net share* - Displays or manages shared printers or directories
- *Net start* - Lists or starts network services
- *Net statistics* - Displays workstation and server statistics
- *Net stop* - Stops services
- *Net time* - Displays or synchronizes network time
- *Net use* - Displays or manages remote connections
- *Net user* - Creates local user accounts
- *Net view* - Displays network resources or computers

## Net User command in Windows Vista

User accounts play a large role in Vista and the "net user" gives a method for managing them. Although user accounts can be managed in several ways through the normal graphical interface, the command line can be quicker and can be scripted.

[Back to top](#)

[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

< [Home](#) | ©2002-2010 Victor Laurie

# The Command Line in Windows

## Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console- Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)

## Netstat

Netstat is a useful tool for checking network and Internet connections. Some useful applications for the average PC user are considered, including checking for malware connections.

### Syntax and switches

The command syntax is

```
netstat [-a] [-b] [-e] [-n] [-o] [-p proto] [-r] [-s] [-v] [interval]
```

A brief description of the switches is given in Table 1 below. *Note that switches for Netstat use the dash symbol "-" rather than the slash "/"*.

Table 1. Switches for Netstat command

Switch	Description
-a	Displays all connections and listening ports
-b	Displays the executable involved in creating each connection or listening port. (Added in XP SP2.)
-e	Displays Ethernet statistics
-n	Displays addresses and port numbers in numerical form
-o	Displays the owning process ID associated with each connection
-p proto	Shows connections for the protocol specified by proto; proto may be any of: TCP, UDP, TCPv6, or UDPv6.
-r	Displays the routing table
-s	Displays per-protocol statistics
-v	When used in conjunction with -b, will display sequence of components involved in creating the connection or listening port for all executables
[interval]	An integer used to display results multiple times with specified number of seconds between displays. Continues until stopped by command <i>ctrl+c</i> . Default setting is to display once,

### Applications of Netstat

Netstat is one of a number of command-line tools available to check the functioning of a network. ([See this page](#) for discussion of other tools.) It provides a way to check if various aspects of TCP/IP are working and what connections are present. In Windows XP SP2, a new switch "-B" was added that allows the actual executable file that has opened a connection to be displayed. This newer capability provides a chance to catch malware that may be phoning home or using your computer in unwanted ways on the Internet. There are various ways that a system administrator might use the assortment of switches but I

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

will give two examples that might be useful to home PC users.

## Checking TCP/IP connections

TCP and UDP connections and their IP and port addresses can be seen by entering a command combining two switches:

```
netstat -an
```

An example of the output that is obtained is shown in Figure 1.

Figure 1. Example output for command "netstat -an"

```
C:\Documents and Settings\Owner>netstat -an
Active Connections
Proto Local Address           Foreign Address         State
TCP    0.0.0.0:135              0.0.0.0:0               LISTENING
TCP    0.0.0.0:445              0.0.0.0:0               LISTENING
TCP    127.0.0.1:1027           0.0.0.0:0               LISTENING
TCP    192.168.1.100:139        0.0.0.0:0               LISTENING
TCP    192.168.1.100:2558      207.68.172.236:80       CLOSE_WAIT
TCP    192.168.1.100:2916      204.14.90.25:21         CLOSE_WAIT
TCP    192.168.1.100:2923      69.65.109.55:80         TIME_WAIT
TCP    192.168.1.100:2924      204.245.162.25:80       ESTABLISHED
TCP    192.168.1.100:2925      66.150.96.119:80       ESTABLISHED
TCP    192.168.1.100:2930      204.245.162.27:80       ESTABLISHED
UDP    0.0.0.0:445              *:.*
UDP    0.0.0.0:500              *:.*
UDP    0.0.0.0:1030             *:.*
UDP    0.0.0.0:1040             *:.*
UDP    0.0.0.0:1155             *:.*
UDP    0.0.0.0:1175             *:.*
UDP    0.0.0.0:4500             *:.*
UDP    127.0.0.1:123            *:.*
UDP    127.0.0.1:1036           *:.*
UDP    127.0.0.1:1900           *:.*
UDP    127.0.0.1:2922           *:.*
UDP    192.168.1.100:123        *:.*
UDP    192.168.1.100:137        *:.*
UDP    192.168.1.100:138        *:.*
UDP    192.168.1.100:1900      *:.*
```

The information that is displayed includes the protocol, the local address, the remote (foreign) address, and the connection state. Note that the various IP addresses include port information as well. An explanation of the different connection states is given in Table II>

Table II. Description of various connection states

State	Description
CLOSED	Indicates that the server has received an ACK signal from the client and the connection is closed
CLOSE_WAIT	Indicates that the server has received the first FIN signal from the client and the connection is in the process of being closed
ESTABLISHED	Indicates that the server received the SYN signal from the client and the session is established
FIN_WAIT_1	Indicates that the connection is still active but not currently being used
FIN_WAIT_2	Indicates that the client just received acknowledgment of the first FIN signal from the server
LAST_ACK	Indicates that the server is in the process of sending its own FIN signal
LISTENING	Indicates that the server is ready to accept a connection
SYN_RECEIVED	Indicates that the server just received a SYN signal from the client
SYN_SEND	Indicates that this particular connection is open and active



TIME\_WAIT

Indicates that the client recognizes the connection as still active but not currently being used

## Checking for malware by looking at which programs initiate connections

To find out which programs are making connections with the outside world, we can use the command

```
netstat -b
```

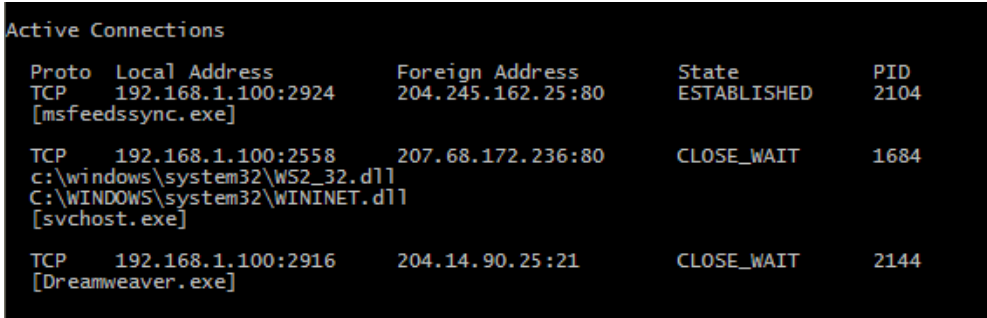
Actually, it is better to check over a period of time and we can add a number that sets the command to run at fixed intervals.

Also, it is best to create a written record of the connections that are made over some period of time. The command can then be written

```
netstat -b 5 >> C:\connections.txt
```

Note that as written, this command will run with five-second intervals until stopped by entering "Ctrl+c", which is a general command to exit. (Some reports say that this can be fairly CPU intensive so it may cause a slower, single-core machine to run sluggishly. It was not noticeable on my dual-core machine.) A simple example of the type of output is shown in Figure 2. Note that the Process ID (PID) is given. This command can be combined with other tools such as [Task Manager](#) to analyze what executable files and processes are active and are trying to make Internet connections.

Figure 2. Sample output for command "netstat -b"



```
Active Connections
Proto Local Address          Foreign Address        State                   PID
TCP   192.168.1.100:2924      204.245.162.25:80     ESTABLISHED            2104
[msFeedssync.exe]
TCP   192.168.1.100:2558      207.68.172.236:80     CLOSE_WAIT             1684
c:\windows\system32\WS2_32.dll
C:\WINDOWS\system32\WININET.dll
[svchost.exe]
TCP   192.168.1.100:2916      204.14.90.25:21       CLOSE_WAIT             2144
[Dreamweaver.exe]
```

## Batch program to check connections and terminate automatically

The previous example of using "netstat -b" to check connections at intervals has the disadvantage that it requires manual termination. It is also possible to use a batch file that runs a specified number of times with a given time interval and then terminates automatically. In Windows XP we can make use of a command from the [Windows 2003 Server Tools](#) called "Sleep".

A possible batch file is:

```
@echo off
for /L %%X in (1,1,100) do (netstat -b >> C:\connections.txt)&(sleep 5)
```

This particular example does 100 iterations of the *netstat* command at 5 second intervals.

[Back to top](#)

# The Command Line in Windows

## Site navigation

[Home](#)
[Assoc and Ftype](#)
[Batch files- basics](#)
[Batch files - branching](#)
[Batch files- iterating](#)
[Command Line-](#)
[Introduction](#)
[Command line list and reference](#)
[Commands that everybody can use](#)
[Configuring the command prompt window](#)
[Doskey](#)
[File system utility- Fsutil](#)
[Net Services](#)
[Netstat](#)
[Network Services Shell](#)
[PowerShell](#)
[Recovery Console](#)
[Recovery Console- Commands](#)
[Registry editor console](#)
[Scripts in the command line](#)
[Server 2003 tools for XP](#)
[Service Controller](#)
[Command \(SC\)](#)
[Shell command](#)
[Start-Run line](#)
[Support tools](#)
[Tasklist](#)
[TCP/IP networking tools](#)
[Tips for using the command shell](#)
[Tskill and Taskkill](#)

## Netsh, the Network Services Shell

A suite of command line networking tools called Netsh that comes with its own shell or interface is contained in a number of Windows operating systems and is discussed here.

### Introduction to Netsh

As more and more home users set up networks, they are finding themselves to be *de facto* system administrators. Home networks are very nice but they require a certain amount of care and feeding. Fortunately, Windows XP comes with a large assortment of command-line tools that can help maintain your network. Although many are specialized and of interest only to administrators of large corporate setups, some tools can be quite helpful to the home user as well.

Many may find that the basic tools like *ping*, *ipconfig*, and *netstat*, which are discussed on [another page](#), are all that they care to deal with but the more adventurous can take advantage of a complete suite of powerful tools called *Netsh*. This suite is invoked from the standard command-line but has its own interface or shell with a large number of sub-commands. I will try to focus on the features of Netsh that I think can be helpful to the home user. The whole suite has many applications and those who want more details can go to this [Microsoft reference](#).

The Network Services shell is opened by entering *netsh* into a regular command prompt. The shell has a hierarchical structure with some sub-shells that Microsoft calls "contexts". From the user's point of view, however, all that this means is that commands are entered as a sequence of terms. The following sections discuss the "contexts" of most use to the home user.

### The "netsh diag" context

The diagnostic context "diag" contains useful tools for checking out a network and testing various components and functions. Table I shows the contexts and sub-commands of most interest to this discussion. A complete list and many details are given at this [Microsoft reference](#).

Table I. Some sub-shells (contexts) and commands for Netsh diag

Context	Sub-context	Commands
diag	<i>connect</i> - Establishes, verifies, and then drops a connection	<i>iphost, mail, news</i>
	<i>ping</i> - Verifies connectivity	<i>adapter, iphost, mail, news</i>
	<i>show</i> - Lists network components and settings	<i>all, client, ip, mail, modem</i>
	<i>gui</i> - Starts the network diagnostics tool in Help and Support Center.	Graphical user interface

This group of commands provides ways to test some of the most common functions of interest to home users. For example, you can test if your email server is working or check your email settings by the command

```
netsh diag connect mail
```

(Note that this may not work for email clients like AOL.). Another example is to list important settings with

```
netsh diag show all
```

[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

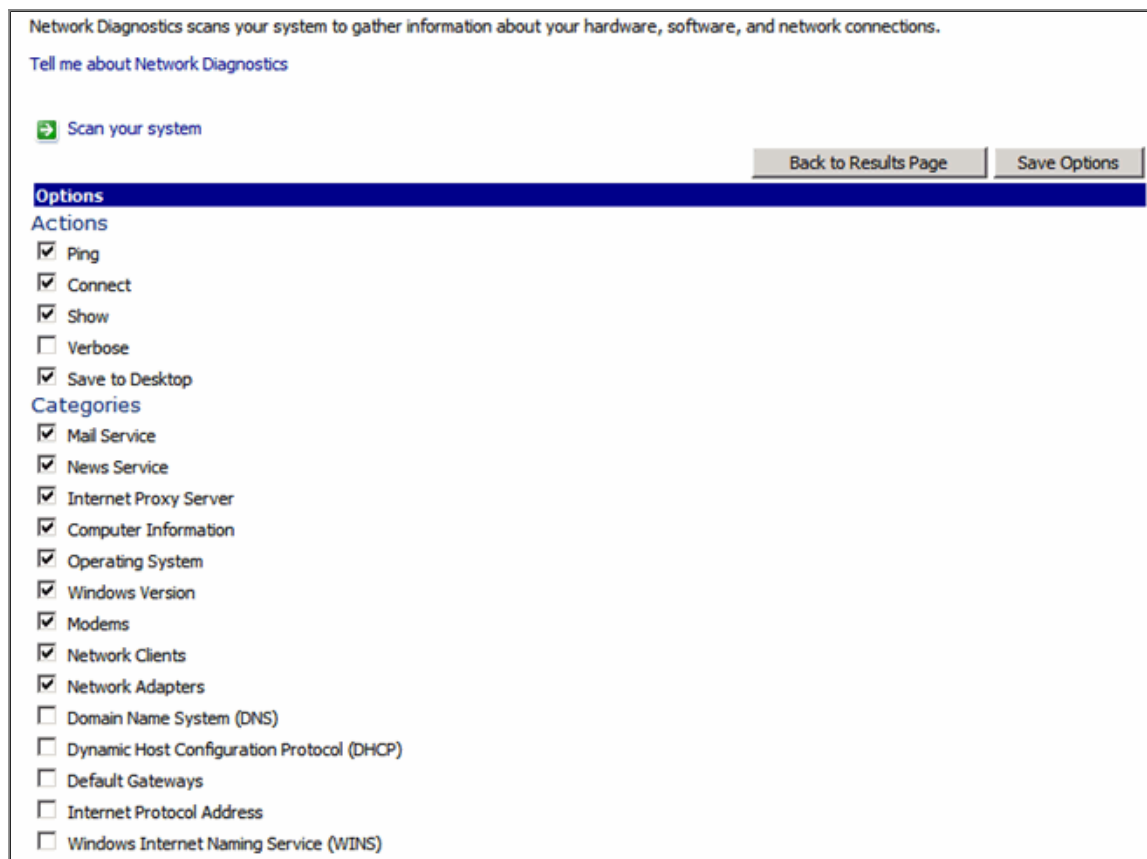
## The graphical user interface

One way to simplify diagnostic tasks is to use the Help Center interface that can be invoked by entering

```
netsh diag gui
```

With this interface, you can carry out a number of diagnostic tests with one operation. Figure 1 shows the available functions.

Figure 1. Settings for GUI function of Netsh diag



## The "netsh interface ip" context

This context is another with functions that might interest a home user. It provides a way to add, delete, modify, and display various IP addresses and TCP/IP settings. Table II lists several functions. More detail and additional functions are discussed in this [Microsoft reference](#). An example of its use is to display TCP/IP settings with the command

```
netsh interface ip show config
```

This can also be written

```
netsh int ip show config
```

Table II. Some commands for "netsh interface ip"

Context	Sub-context	Function
<i>interface ip</i> or <i>int ip</i>	<i>set address</i>	Configures an IP address and a default gateway on a specified interface
	<i>show address</i>	Displays address for specified interface

## Reset Internet Protocol (TCP/IP)

Another example of using the *Netsh Internet Ip* context is resetting TCP/IP. For several reasons, including infestation from

spyware, these settings might get corrupted. *Netsh* contains a command that will reset the TCP/IP stack to the original settings that existed when the operating system was installed. These settings may not be the most up-to-date for your system but they will allow you to reconfigure after a spyware invasion or other problem. The command to reset TCP/IP is

```
netsh int ip reset {logfile}
```

Note that you must include the name of a file where a log of the actions will be placed. Thus, if the log file is *C:\tcplog.txt*, the command is

```
netsh int ip reset C:\tcplog.txt
```

A detailed description of reinstalling TCP/IP is given at this [Microsoft reference](#). Also, see the Winsock section below.

## The "netsh winsock" context

Service pack 2 for Windows XP includes some additions to the Netsh suite. These are [discussed here](#) and include a new tool for repairing the Windows network socket or "winsock". Uninstalling spyware or poorly written applications can corrupt the winsock settings and result in loss of network connectivity. There are two commands for managing the settings. To display a list of various installed services (LSP, BSP, NSP) use

```
netsh winsock show catalog
```

This list may not be too meaningful for the average PC user but it can be helpful for more experienced users. To reset the winsock to the default configuration use

```
netsh winsock reset catalog
```

Note that any installed software that uses Internet connections may be partially disabled by this action and have to be reinstalled. This would include most software that updates itself and anti-virus programs that use proxy servers. Therefore, this command should only be used for cases where the degree of corruption makes it necessary. See [this reference](#) for some alternative methods.

## Netsh Firewall Helper in Windows XP SP2

Microsoft changed the way the firewall in Windows XP works when it issued service pack 2. It also added capability to Netsh for extensive configuring of the firewall with a new context "netsh firewall" that Microsoft calls the Firewall Helper. Its use in troubleshooting firewall problems in SP2 is extensively discussed [in this knowledge base article](#). With the Firewall Helper Microsoft says you can now

- *Configure the default state of Windows Firewall. (Options include Off, On, and On with no exceptions.)*
- *Configure the ports that must be open.*
- *Configure the ports to enable global access or to restrict access to the local subnet.*
- *Set ports to be open on all interfaces or only on a specific interface.*
- *Configure the logging options.*
- *Configure the Internet Control Message Protocol (ICMP) handling options.*
- *Add or remove programs from the exceptions list*

The number of possible commands is quite large but two main sub-contexts are

```
netsh firewall set
```

and

```
netsh firewall show
```

An extensive list of commands is [in the knowledge base article](#) previously mentioned.

[Back to top](#)

# The Command Line in Windows

## Site navigation

[Home](#)
[Assoc and Ftype](#)
[Batch files- basics](#)
[Batch files - branching](#)
[Batch files- iterating](#)
[Command Line-](#)
[Introduction](#)
[Command line list and reference](#)
[Commands that everybody can use](#)
[Configuring the command prompt window](#)
[Doskey](#)
[File system utility- Fsutil](#)
[Net Services](#)
[Netstat](#)
[Network Services Shell](#)
[PowerShell](#)
[Recovery Console](#)
[Recovery Console- Commands](#)
[Registry editor console](#)
[Scripts in the command line](#)
[Server 2003 tools for XP](#)
[Service Controller](#)
[Command \(SC\)](#)
[Shell command](#)
[Start-Run line](#)
[Support tools](#)
[Tasklist](#)
[TCP/IP networking tools](#)
[Tips for using the command shell](#)
[Tskill and Taskkill](#)

## PowerShell in Windows XP

Microsoft has introduced a entirely new command line interface called "PowerShell". Some of the new features are described.

### Introduction to PowerShell

Microsoft originally intended to develop a new command line interface (first called "Monad") as part of Vista. However, it was subsequently decided to make the feature (renamed "PowerShell") a stand-alone application and it is [now available](#) for Windows XP SP2. (The Vista version has now been released and is [available at this site](#). Incidentally, Vista still has the command interpreter *cmd.exe* just like Windows XP.) The new shell is Microsoft's answer to Unix shell scripting. It is quite different from the previous command line interface and is considerably more powerful. It makes use of more sophisticated techniques and [objects](#) and requires the .NET Framework 2.0. It has new functions for systems and network administration and is aimed at IT professionals. Because the purpose of this present site is to introduce the command line to home PC users , PowerShell is somewhat beyond the intended scope (and my personal experience). Nonetheless, I believe the home user should be aware of PowerShell's potential and the more experienced may wish to explore it further. I will try to outline very briefly what PowerShell is about.

### PowerShell features

In the previous Windows command line described elsewhere on this site, commands consist of internal command strings that are interpreted and executed by the command interpreter or of commands that invoke separate executable files. PowerShell has a new approach that makes use of what Microsoft calls "cmdlets". Here is Microsoft's description:

*A cmdlet (pronounced "command-let") is a single-feature command that manipulates objects in Windows PowerShell. You can recognize cmdlets by their name format -- a verb and noun separated by a dash (-), such as Get-Help, Get-Process, and Start-Service.*

Although each cmdlet has a single function, groups of cmdlets can be strung together to carry out a complex task. Also the output of many cmdlets can be used as input (piped) to other cmdlets without additional processing. These capabilities represent a significant advance over the present command line shell.

PowerShell continues to recognize the commands from the older command shell although, in many cases, the command is an alias for a PowerShell cmdlet

### List of cmdlets

At this time, PowerShell comes with 129 cmdlets. Since cmdlets are easily written, more can be expected. Table I shows the list of those presently available.

Table I. List of PowerShell cmdlets

Add-Content	Get-Date	Move-ItemProperty	Set-AuthenticodeSignature
Add-History	Get-EventLog	New-Alias	Set-Content
Add-Member	Get-ExecutionPolicy	New-Item	Set-Date

[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

Add-PSSnapin	Get-Help	New-ItemProperty	Set-ExecutionPolicy
Clear-Content	Get-History	New-Object	Set-Item
Clear-Item	Get-Host	New-PSDrive	Set-ItemProperty
Clear-ItemProperty	Get-Item	New-Service	Set-Location
Clear-Variable	Get-ItemProperty	New-TimeSpan	Set-PSDebug
Compare-Object	Get-Location	New-Variable	Set-Service
ConvertFrom-SecureString	Get-Member	Out-Default	Set-TraceSource
Convert-Path	Get-PfxCertificate	Out-File	Set-Variable
ConvertTo-Html	Get-Process	Out-Host	Sort-Object
ConvertTo-SecureString	Get-PSDrive	Out-Null	Split-Path
Copy-Item	Get-PSPProvider	Out-Printer	Start-Service
Copy-ItemProperty	Get-PSSnapin	Out-String	Start-Sleep
Export-Alias	Get-Service	Pop-Location	Start-Transcript
Export-Clixml	Get-TraceSource	Push-Location	Stop-Process
Export-Console	Get-UiCulture	Read-Host	Stop-Service
Export-Csv	Get-Unique	Remove-Item	Stop-Transcript
ForEach-Object	Get-Variable	Remove-ItemProperty	Suspend-Service
Format-Custom	Get-WmiObject	Remove-PSDrive	Tee-Object
Format-List	Group-Object	Remove-PSSnapin	Test-Path
Format-Table	Import-Alias	Remove-Variable	Trace-Command
Format-Wide	Import-Clixml	Rename-Item	Update-FormatData
Get-Acl	Import-Csv	Rename-ItemProperty	Update-TypeData
Get-Alias	Invoke-Expression	Resolve-Path	Where-Object
Get-AuthenticodeSignature	Invoke-History	Restart-Service	Write-Debug
Get-ChildItem	Invoke-Item	Resume-Service	Write-Error
Get-Command	Join-Path	Select-Object	Write-Host
Get-Content	Measure-Command	Select-String	Write-Output
Get-Credential	Measure-Object	Set-Acl	Write-Progress
Get-Culture	Move-Item	Set-Alias	Write-Verbose
			Write-Warning

## PowerShell Cmdlet syntax

There are a number of parameters possible for cmdlets and a detailed discussion of syntax is beyond our scope. I will try to hint at the range of possibilities by discussing one useful cmdlet that carries out the copying function. It is not limited to copying files and folders but can also copy Registry keys and entries. This one cmdlet, in fact, incorporates the functions of several older commands with greater flexibility. First, here is a simple example where a folder and all its contents are to be copied

```
Copy-Item C:\Logfiles -destination D:\Backup -recurse
```

This cmdlet copies all files and sub-folders in the folder C:\Logfiles to the folder D:\Backup. The parameter "-recurse" is used when sub-folders are to be copied.

Next, here are all the parameters in their full glory:

```
Copy-Item [-path] <string[]> [[-destination] <string>] [-container] [-recurse] [-force]
[-include <string[]>] [-exclude <string[]>] [-filter <string>] [-passThru] [-credential
<PSCredential>] [-whatIf] [-confirm] [<CommonParameters>]
```

Naturally, the full set of parameters varies from one cmdlet to the next but one option that is common to many is the intriguing "-"

`whatIf`". This setting describes what would happen if you executed the command but without actually executing it. This allows you to see safely what would happen if you did the command. For a table describing the various parameters above, [click here](#).

## PowerShell Scripting

PowerShell is also the basis for a scripting language. An overview of the available operators and functions is at [this MSDN reference](#). This language is intended to make administrative tasks easier and seems likely to supplant VBScript in the future. The extension for PowerShell scripts is `.PS1`. Many security features are built into the scripting engine and the default setting is to prevent scripts from running. Permission to run scripts is controlled by a feature called "Execution Policy". Information about this feature can be obtained by the PowerShell command

```
Get-Help about_signing
```

More about PowerShell scripting can be found at [this Microsoft site](#).

## More information

We can barely scratch the surface in this very short description of PowerShell. For those who wish to explore the subject further, here are some references:

- [Microsoft main page on PowerShell](#)
- [Wikipedia article](#)
- [PowerShell Wiki](#)

[Back to top](#)



# The Command Line in Windows

## Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console- Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the command shell](#)

[Tskill and Taskkill](#)

## The Recovery Console in Windows XP

A special example of a command-line accessory is the Windows XP Recovery Console, which can be a great help if Windows needs repairing and won't boot, especially for NTFS machines.

Even if you never intend to use the command line, it would pay to become familiar with the Recovery Console. In the event of a problem getting Windows XP to boot, this command line facility might allow you to repair a corrupted system or to retrieve precious personal files. In FAT systems, a [DOS boot disk](#) can be used to access a system with problems but the NTFS systems that are now common cannot be accessed from DOS. As outlined on [another page](#), various CD boot disks can be constructed that provide for emergency services either with Linux or special NTFS readers. However, Windows XP comes with its own accessory for this purpose. The Recovery Console will work with both FAT and NTFS formatted disks but is especially useful for systems with NTFS. It is not as well known as it should be because Microsoft seems to want to keep it obscure and hard to use. In this article, I will try to clarify its usage and explain how it can help the average PC owner.

### Accessing the Recovery Console with a Windows CD

If you have a full Windows disk, the console can be accessed by booting from the Windows XP CD. It will take its time loading the setup but eventually you will see the "Welcome to Setup" screen. Enter the letter "r" and a prompt will appear asking for the administrator password. Users of the Home edition or those without an administrator password can just enter a blank. (Note that this password is not necessarily the same as the one for a user account, even if that account has administrator privileges). The screen will then show a numbered list of all your Windows installations (most people will have only one.) It will ask, "Which Windows installation would you like to log onto?" Unless you have a multi-boot system, enter the number "1". You must enter a number. Do NOT just press the Enter key. Some time may pass but eventually the Recovery Console will load and provide a command prompt where various system tasks can be performed.

### Accessing the Recovery Console Without a Windows CD

Computer vendors very often do not provide a Windows disk with their systems. Instead they provide a [Restore disk](#) or just a hidden restore partition. They *may* provide a way to get into the Recovery Console but computer owners will have to check their particular setup. If, as is likely, the Recovery Console is missing there is another way to obtain it.

Microsoft provides software for creating a set of six diskettes for reinstalling Windows XP for those with no bootable CD. The [details are here](#). It doesn't seem to be widely known but these disks also contain the Recovery Console. Thus creating this set of diskettes will give those without a Windows CD (but with a floppy drive) the option of using the Recovery Console. It is tedious; you have to wait while the system slogs through the setup process and loads all six disks but if it saves your system it will be worth it. Once you get to the Welcome Screen, the procedure is the same as with that with a CD described above.

If you have neither a Windows XP CD nor a floppy drive, there are methods for placing floppy disk images on a bootable CD. One easy way to create a bootable CD containing the Recovery Console from the Microsoft file mentioned above has been provided by Dean Adams and can be [downloaded at this link](#).

### Commands in the Recovery Console

[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

Once you have opened the Recovery Console, what can you do with it? Table I lists commands that are available. Many have switches. Details about syntax and functions are available [on a following page](#) or can be obtained by entering

```
{command} /?
```

in the Recovery Console. Details are also available on [this Microsoft page](#). Note that the console has its own command interpreter and commands may differ somewhat from those of the same name in a regular command prompt. Also, in the default configuration, some commands are disabled or limited in their functions. Removing some of the restrictions is discussed in the next section.

[Back to the top](#)

Table I. Available commands in the Recovery Console

Command	Description
Attrib	Changes the attributes of a file or directory
Batch	Executes the commands specified in the text file
Bootcfg	Boot file (boot.ini) configuration and recovery
ChDir (Cd)	Displays the name of the current directory or changes the current directory
Chkdsk	Checks a disk and displays a status report
Cls	Clears the screen
Copy	Copies a single file to another location
Delete (Del)	Deletes one or more files
Dir	Displays a list of files and subdirectories in a directory
Disable	Disables a system service or a device driver
Diskpart	Manages partitions on your hard drives
Enable	Starts or enables a system service or a device driver
Exit	Exits the Recovery Console and restarts your computer
Expand	Extracts a file from a compressed file
Fixboot	Writes a new partition boot sector onto the specified partition
Fixmbr	Repairs the master boot record of the specified disk
Format	Formats a disk
Help	Displays a list of the commands you can use in the Recovery Console
Listsvc	Lists the services and drivers available on the computer
Logon	Logs on to a Windows installation
Map	Displays the drive letter mappings
Mkdir (Md)	Creates a directory
More	Displays a text file
Rename (Ren)	Renames a single file
Rmdir (Rd)	Deletes a directory

Set	Displays and sets environment variables
Systemroot	Sets the current directory to the systemroot directory of the system you are currently logged on to.
Type	Displays a text file

[Back to the top](#)

## Removing Console Restrictions

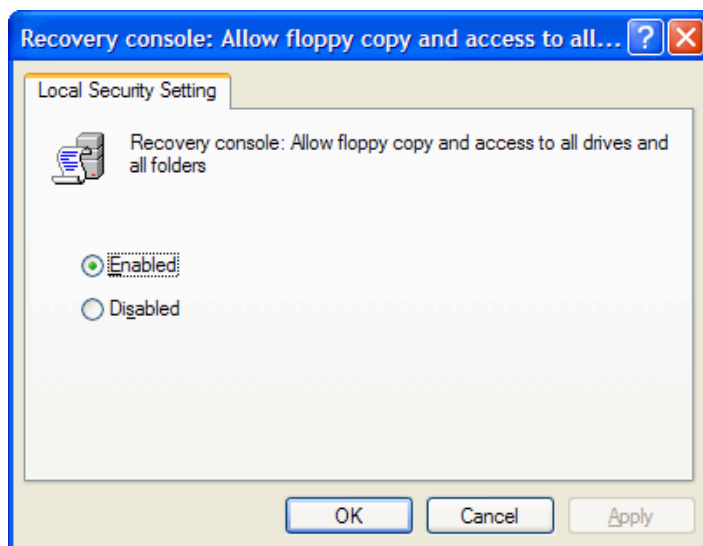
The default configuration of the console contains a number of limitations. I presume the rationale behind this is to limit the damage that the uninitiated might do. However, it greatly reduces the usefulness of the console. The restrictions include denial of access to any but a limited number of folders, inability to write to a floppy disk, and inability to use wild cards in commands. These restrictions can be removed by several methods. They are described in the next three sections. Note that these measures must be taken from a regular Windows logon with administrator privileges. They cannot be made from the Recovery Console itself.

### Using the Group Policy or Security Policy Management Consoles

This method is open to Windows XP Professional users only and involves a section of the Group Policy Editor called Local Security Settings. A description is given at this [Microsoft article](#) but it is not a model of clarity. (Microsoft instructions in this area are confusing in general.) There are two possible management consoles that can be used, the Group Policy editor *gpedit.msc* or its subsection *secpol.msc* (See this reference for a [discussion of management consoles](#).) The simplest way is to open the Local Security Settings console by entering

```
secpol.msc
```

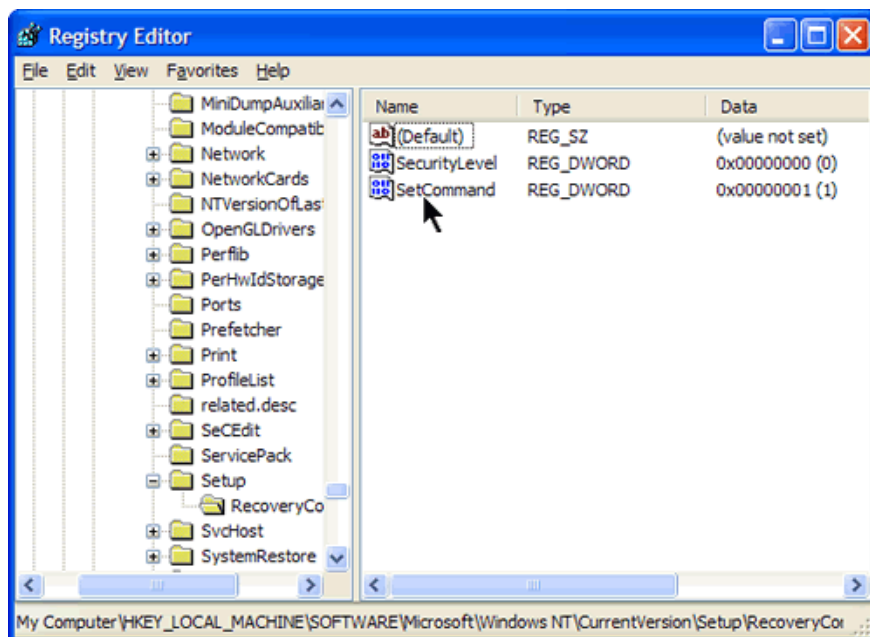
into Start-Run. Then click "Local Policies-Security Options" in the left pane. In the list that appears in the right pane find, Recovery Console: Allow Floppy Copy And Access To All Drives And Folders, and double-click it. The box shown in the figure below will open. Click the button "Enabled" and then "OK"



The name of this setting is misleading. There is more involved than enabling writing to a floppy or access to all folders. What this actually does is to give you permission to use the "Set" command within the Recovery Console. This command is discussed later and is what is actually used to remove restrictions.

### Editing the Registry

The procedure described above works by changing an entry in the Registry. If you are familiar with Registry editing, direct editing is another route and it has the advantage of being available to those with the Home Edition of Windows XP. Open *regedit* and find HKEY\_LOCAL\_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Setup\RecoveryConsole. In the right pane, this key will have an entry "SetCommand". Put a value of 1. This is shown in the figure below. *Be sure to back up the Registry before doing any editing.*



## A script for doing the Registry edit

Editing the Registry is not an activity for everyone. For the benefit of those who are uncertain about editing the Registry themselves, I have created a small script that will make the Registry change for you. For this script, I have chosen the INF format. You may be familiar with INF files since they are the standard way to install drivers. [Click here to download setcommand.inf](#) (zipped). Before using it, back up the Registry or make a Restore Point. Unzip the file anywhere that is convenient and then right-click once to open the context menu. Click the menu item "Install". That's all there is to it. Now your Registry has been edited. Do not use this file on Windows 98/Me systems.

## Using the "Set" Command

The Registry edit discussed in the sections above does not actually allow you any new functions until one more step is done within the the Recovery Console itself. The Registry edits do not make changes. They give you *permission* to make changes using the "set" command. Those who are familiar with the regular set command line will be familiar with using "set" and environment variables. (More on this subject [can be read here](#).) The capacities of the Recovery Console can be expanded with four "set" commands. The first allows access to all files and folders on the main system. In the Recovery Console prompt enter

```
set AllowAllPaths = true
```

The second command allows you to copy files to removable media like floppy disks. Enter

```
set AllowRemovableMedia = true
```

The next command allows you to use wildcards in commands like "dir" and "del". Enter

```
set AllowWildCards = true
```

Finally, there is a command that allows you to copy files without being prompted to continue when you are overwriting an existing file. Enter

```
set NoCopyPrompt = true
```

There must be a space before and after the "equals" sign in all the set commands or they won't work. The commands are not

case-sensitive.

## The One Command to Learn

If you learn no other commands to use in the Recovery Console, learn this:

```
chkdsk C: /r
```

There is more than one report on the Web of how this command fixed a system that wouldn't boot. If you are using the console from the floppy disk installation, *chkdsk* will complain that it can't locate the file *autochk.exe*. When it asks for that file's location, point it to `\windows\system32`. This command will thoroughly examine your hard drive and is not a fast process so give it time.

[Back to the top](#)

# The Command Line in Windows

## Recovery Console Commands

The command shell that is available in the Recovery Console differs from the command prompt in the normal Windows XP operating system. The available commands are discussed in this article.

A [previous page discussed the Recovery Console](#) and the commands that are available. In this article are some additional details about the commands.

### Attrib

Changes the attributes of a file or directory. Has different parameters from the standard command prompt. The syntax is:

```
attrib [+r|-r] [+s|-s] [+h|-h] [+c|-c] [[drive:][path] filename]
```

The switches are described in the table below:

Switch	Function
r	Controls "read-only" attribute
s	Controls "system" attribute
h	Controls "hidden" attribute
c	Controls compressed file attribute

### Batch

Executes the commands specified in a text file. If desired, the results can be sent to an output file. Not available except when using the Recovery Console. The Syntax is:

```
batch input_file.txt [output_file]
```

### Bootcfg

Used to configure boot file *boot.ini*. Has different parameters from the standard command prompt. Syntax is:

```
bootcfg /parameter
```

The possible parameters are given in the table below.

Parameter	Function
add	Add a Windows installation to the boot list
copy	Back up <i>boot.ini</i>
default	Choose the default boot entry
disabledredirect	Disable redirection in the boot loader
list	List the current entries in the boot list
rebuild	Iterate through all Windows installations
redirect	Enable redirection in the boot loader

## Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)

[Vista command list](#)

scan	Scan all disks for Windows installations
------	--

[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

### CD (Chdir)

Displays the name of the current directory or changes the current directory. No switches. Behaves somewhat differently from command prompt.

### Chkdsk

Checks a disk and displays a status report. Has different parameters from the standard command prompt. Syntax is:

```
chkdsk [drive:] [/p] [/r]
```

The switch /p makes an exhaustive check of a drive without making changes. The switch /r makes the same check but also recovers readable information. (Note that there are erroneous entries on the Internet that suggest using switches that are only available in the regular command prompt.)

### Cls

Clears the screen. Has same function as in the command prompt.

### Copy

Copies a single file to another location. Lacks the switches present in the command prompt.

### Del

Deletes a single file. Lacks the switches present in the command prompt.

### Dir

Displays a list of the files and subdirectories in a directory. Lacks the switches present in the command prompt.

### Disable

Disables a system service or a device driver. The command is only available when you are using the Recovery Console.

The syntax is:

```
disable {[service_name] | [device_driver_name]}
```

### Diskpart

Creates and deletes partitions on a hard drive. The command is different from the very restricted version available in a normal command prompt. The syntax is:

```
diskpart [/add | /delete] [device_name | drive_name | partition_name] [size]
```

### Enable

Starts or enables a system service or a device driver. The command is only available when you are using the Recovery Console. The syntax is:

```
enable {service_name | device_driver_name} [startup_type]
```

### Exit

Exits the Recovery Console and restarts your computer.

### Expand

Extracts a file from a compressed file. Has switches that differ from the command prompt version. The syntax is:

```
expand source [/F:filename] [destination] [/d] [/y]
```

The switch /F:filename allows you to extract a single file indicated by *filename*. To list the files contained in a cabinet file without extracting them, use the switch /d. The switch /y suppresses prompts when over-writing a file with an extracted file.

### Fixboot

Writes a new partition boot sector to the system partition. The fixboot command is only available when you are using the Recovery Console. Syntax is:

```
fixboot [drive]
```

### Fixmbr

Repairs the master boot record of the specified disk. The fixmbr command is only available when you are using the Recovery Console. The syntax is:

```
fixmbr [device_name]
```



If you do not specify a *device\_name*, a new master boot record will be written to the boot device, which is the drive on which your primary system is loaded.

#### Format

Formats the specified drive to the specified file system. The syntax is:

```
format [drive:] [/q] [/fs:file-system]
```

The switch /q enables a quick format. The switch /fs: allows a choice of file system.

#### Help

Displays a list of the commands you can use in the Recovery Console.

#### Listsvc

Lists the services and drivers available on the computer. Only available when you are using the Recovery Console.

#### Logon

Logs on to a Windows installation. Only available when you are using the Recovery Console.

#### Map

Displays the drive letter mappings. Only available when you are using the Recovery Console.

#### Md (mkdir)

Creates a directory or subdirectory.

#### More

Displays a text file.

#### Net use

Connects a network share to a drive letter. The net use command with different parameters is available from the command prompt.

#### Rd (rmdir)

Removes (deletes) a directory. Lacks the switches available in the command prompt.

#### Ren (rename)

Changes the name of a single file.

#### Set

Displays and sets Recovery Console environment variables. [Details are given on a previous page.](#)

#### Systemroot

Sets the current directory to the systemroot folder of the Windows installation where you are logged on. Not available in the command prompt.

#### Type

Displays a text file.

[Back to top](#)

# The Command Line in Windows

## Site navigation

[Home](#)
[Assoc and Ftype](#)
[Batch files- basics](#)
[Batch files - branching](#)
[Batch files- iterating](#)
[Command Line-](#)
[Introduction](#)
[Command line list and reference](#)
[Commands that everybody can use](#)
[Configuring the command prompt window](#)
[Doskey](#)
[File system utility- Fsutil](#)
[Net Services](#)
[Netstat](#)
[Network Services Shell](#)
[PowerShell](#)
[Recovery Console](#)
[Recovery Console-](#)
[Commands](#)
[Registry editor console](#)
[Scripts in the command line](#)
[Server 2003 tools for XP](#)
[Service Controller](#)
[Command \(SC\)](#)
[Shell command](#)
[Start-Run line](#)
[Support tools](#)
[Tasklist](#)
[TCP/IP networking tools](#)
[Tips for using the](#)
[command shell](#)
[Tskill and Taskkill](#)

## Managing the Windows Registry from the Command Prompt with Reg.exe

The command-line utility **reg.exe** is a powerful and versatile way to manage the Windows XP Registry. This article discusses Its features and application.

Many will be familiar with the graphical interface tool *regedit.exe* that is available for editing the Windows Registry. Less familiar, however, is the command-line utility *reg.exe* that also comes with Windows XP. This accessory will do anything that *regedit.exe* can do and has the additional facility of being directly usable in scripts. It is a common tool for system administrators with many computers to manage but can also be useful to the more experienced home PC user. I will discuss some aspects that may be of interest to this latter group. More details can be found at this [Microsoft site](#). There is also information in the Windows XP *Help and Support Center*.

Registry editing is not for everybody but it is not as fearsome an operation as it is sometimes made out to be. Just be sure to follow the iron-clad rule to back up the Registry first before editing. There are many useful tweaks that involve a simple Registry edit and *reg.exe* provides a way that is simpler and safer in some ways than Regedit. It also provides a way to back up keys or entire hives of the Registry into files that can be stored off the main drive.

Like some other command-line utilities, the reg command is a shell or console that has its own set of sub-commands. An complete command will consist of

```
reg subcommand variables
```

Table I lists these subcommands and some are discussed in more detail in sections that follow. The commands can be carried out on remote networked computers as well as the local computer but I will confine the discussion to operations involving just the local computer.

Table I. Subcommands for reg.exe

Subcommand	Function
add	Adds a new subkey or entry to the registry
delete	Deletes a subkey or entries from the registry
query	Displays the data in a subkey or a value
compare	Compares specified registry subkeys or entries
copy	Copies a subkey to another subkey.
save	Saves a copy of specified subkeys, entries, and values of the registry in hive (binary) format
restore	Writes saved subkeys and entries in hive format back to the registry
load	Writes saved subkeys and entries in hive format back to a different subkey
unload	Removes a section of the registry that was loaded using reg load
export	Creates a copy of specified subkeys, entries, and values into a file in REG (text) format
import	Merges a REG file containing exported registry subkeys, entries, and values into the registry

[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

[Back to top](#)

## Reg add

This command is used to add keys and values to the Registry. The syntax is given by

```
REG ADD KeyName [/v ValueName | /ve] [/t Type] [/s Separator] [/d Data] [/f]
```

Table II explains the entries.

Table II. Parameters in REG ADD command

Parameter	Description
KeyName	Complete Registry key name. Uses abbreviations HKCR, HKCU, HKLM, and HKU for root keys
/v ValueName	Adds or changes a value
/ve	Changes a key's default value
/t Type	The type of value: REG_BINARY, REG_DWORD, REG_SZ, REG_MULTI_SZ, etc. The default is REG_SZ
/s Separator	Specifies the character used to separate strings in REG_MULTI_SZ entries. The default is /0
/d Data	The data to assign to a value
/f	Forces overwriting of existing values with prompting

[Back to top](#)

REG ADD provides a quick and simple method for adding new keys to the Registry or modifying old ones. As an example, let's look at how to add the sub-key "HackersAreUs" to the Local Machine Software key. The command would be

```
REG ADD HKLM\Software\HackersAreUs
```

Now let's add a value named "Stuff" and make it a binary entry with data "0001". The command would be

```
REG ADD HKLM\Software\HackersAreUs /v Stuff /t REG_BINARY /d 0001
```

The two commands could have been executed as a single command but I have split them to make the process clearer. I have used upper case for REG ADD but that is for clarity and is not required.

## Reg delete

Keys and values can be deleted in a similar but somewhat simpler fashion. The syntax is

```
REG DELETE KeyName [/v ValueName | /ve | /va] [/f]
```

Table III describes the parameters.

Table III. Parameters in REG DELETE command

Parameter	Description
KeyName	Complete Registry key name. Uses abbreviations HKCR, HKCU, HKLM, and HKU for root keys
/v ValueName	Deletes a value
/ve	Deletes a key's default value

/va	Deletes all values from a key
/f	Forces deletion with prompting

[Back to top](#)

## Backing up and restoring the Registry

Providing methods for backing up and restoring the Registry are some of the most important applications for *regedit.exe*. There are two file formats that can be used, either a binary format known as a *hive* file or a special text format known as a REG file. The latter format may be more familiar since it is often used for Registry tweaks. The relevant commands are discussed in the following sections.

### Saving and restoring hive files

To create a binary backup, use the command

```
REG SAVE KeyName FileName
```

Hive files are better for backup than REG files because they completely replace the contents of a key when they are restored.

The restore command is

```
REG RESTORE KeyName FileName
```

### Exporting and importing REG files

REG files are specially formatted text files with the extension "reg" that are copies of one or more Registry keys. They are often encountered as a way to carry out small Registry edits or in using Regedit.. They can also be used for backup. The commands are

```
REG EXPORT KeyName FileName
```

and

```
REG IMPORT FileName
```

Note that when a REG file is imported it, it is *merged* with Registry entries rather than completely replacing them. Values that the REG file does not contain are not removed.

## Reg Query

If you want to take a quick look at what is contained in a particular Registry key or in a particular value, you can use the command

```
REG QUERY KeyName [/v ValueName | /ve] [/s]
```

The only new parameter here is */s*. This switch will cause all the subkeys and values in a key to be queried.

[Back to top](#)

# The Command Line in Windows

## Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)

## Running VBScript and JScript files from the Command Shell

Using the Windows script host to run scripts from the command line is discussed. Some specific examples are given.

Windows XP comes with two interfaces for running VBScript and JScript (Microsoft's version of JavaScript) files. The default is a graphical user interface using *Wscript.exe*. However, there is also a command-line interface, *Cscript.exe*. It is often more convenient to use the command line for scripts and Windows XP actually comes with a number of useful scripts for system administration that need to be run from the command prompt.

### Configuring the command shell to run scripts

Many scripts for computer management are more conveniently run from the command shell so that the user does not have to deal with the complications of windows and dialog boxes from the graphical user interface. It can be easier to handle output in a command window. One way to run a script in the command line is to preface the script with the executable *Cscript.exe*. For example, a statement of the form

```
cscript.exe somescript.vbs
```

can be entered into the command line and *somescript.vbs* will then run in the command window. Note, however, that the fully qualified path must generally be used and pathnames with spaces must be enclosed in quotation marks. To make the script processor Cscript the default host for scripts, enter into the command line

```
wscript //H:cscript
```

Scripts can then be run by simply entering into the command line

```
somescript.vbs
```

The default can be returned to the graphical interface with the command

```
wscript //H:wscript
```

### Some scripts that come with Windows XP

Tucked away in the folder `\Windows\System32\` are some files in VBScript format that most PC users have never heard of. Also included with the [two system tool](#) packages discussed on [other pages](#) are some tools that are in the form of scripts. Some of these are really just for administrators but there are several that might be of use to the average PC owner. They are listed below. More detail about these scripts is in the Windows XP Help and Support Center. Running these scripts is easier if Cscript is made the default interface.

Eventquery.vbs

Lists the events and event properties from one or more event logs. Can be used with a filter to specify the types of events to include in or exclude from the query.

Pagefileconfig.vbs

Enables an administrator to display and configure a system's paging file Virtual Memory settings

Prncnfg.vbs

Configures or displays configuration information about a printer. Used without parameters, *prncnfg.vbs* displays command-line help.

Prndrvr.vbs

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

Adds, deletes, and lists printer drivers. Used without parameters, *prndrvr.vbs* displays command-line help.

Prnjobs.vbs

Pauses, resumes, cancels, and lists print jobs. Used without parameters, *prnjobs.vbs* displays command-line help.

Prnmngr.vbs

Adds, deletes, and lists printers or printer connections, in addition to setting and displaying the default printer. Used without parameters, *prnmngr.vbs* displays command-line help.

Prnport.vbs

Creates, deletes, and lists standard TCP/IP printer ports, in addition to displaying and changing port configuration. Used without parameters, *prnport.vbs* displays command-line help.

Prnqctl.vbs

Prints a test page, pauses or resumes a printer, and clears a printer queue. Used without parameters, *prnqctl.vbs* displays command-line help.

## Windows Management Instrumentation Command-line (WMIC) tool

WMIC is a command-line and scripting interface that simplifies the use of Windows Management Instrumentation (WMI). WMIC is based on aliases. Aliases make the primary data provided by WMI available without having to understand WMI-specific concepts. More details are at [this Microsoft reference](#). Information is also available on a local computer by entering into a command prompt

```
WMIC / ?
```

One use of WMIC is to write simple scripts to automate the management of a computer.

[Back to top](#)



# The Command Line in Windows

## Useful Tools for Windows XP from the Server 2003 Resource Kit

Many of the command-line tools from the Windows Server 2003 resource kit can be used in Windows XP. Some are described here.

Microsoft provides a large assortment of command-line tools, which can be obtained in a variety of ways. For users of Windows XP Professional, there are three major sources. One set of "native" tools is part of the standard installation and these are listed [on another page](#). A second set, called "support tools", can be installed from a full version of Windows XP Professional or downloaded. These are the subject of [this page](#) on the present site. A third set is part of the Windows Server 2003 Resource Kit but are applicable to Windows XP. They can be [downloaded here](#) and are the subject of this article.

As would be expected, the Windows 2003 Server tools are intended for large networked systems and many are not appropriate for typical home systems. Nonetheless, many can be useful in a simple system. I have made a somewhat arbitrary selection of those that seem most helpful and discuss them below. Note that some of these tools are actually graphical (GUI) tools that are opened from the command line and this is indicated where it applies. A full list of the tools is available at the Microsoft [download site](#). Once they are installed, the Help and Support Center will contain a list with descriptions and syntax for their use.

### Cdburn.exe: ISO CD-ROM Burner Tool

CDBurn is a command-line tool that allows the user to write data images from image files located on the hard drive to a CD. The data image can be any kind of data, even raw data. This tool can also be used to erase CD-RW media.

### Chklnks.exe: Link Check Wizard

Link Check Wizard (ChkLnks) is a GUI tool that scans all the link (shortcut) files on a computer to determine whether or not the shortcuts point to existing applications or documents. When Link Check Wizard does not find an associated application or document, the wizard lists that file as a dead link, giving you the option to remove it.

### Cleanspl.exe: Spooler Cleaner

Spooler Cleaner (CleanSpl) is a GUI tool that restores the print spooler to its original state. It deletes all print jobs, printers, printer drivers, and spool files on a specified local or remote server.

### Creatfil.exe: Create File

CreatFil creates a blank file of a specified size, filled with space characters.

### Diskraid.exe: RAID Configuration Tool

DiskRaid is a command-line tool that enables configuration and management of redundant array of independent (or inexpensive) disks (RAID) storage subsystems.

### Dvdburn.exe: ISO DVD Burner Tool

DVDBurn is a command-line tool that allows the user to write DVD images from image files located on the hard drive to DVD media. The data image can be created by any program capable of producing DVD image files. You can also use this tool to erase DVD media.

### Empty.exe: Free Working Set Tool

Free Working Set Tool (Empty) is a command-line tool that frees the working set of a specified task or process, making those page frames available for other processes.

### Iniman.exe: Initialization Files Manipulation Tool

IniMan is a command-line tool that enables you to add, delete, modify, or query sections or keys in an .ini file.

### Instsrv.exe: Service Installer

## Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and](#)

[reference](#)

[Commands that everybody](#)

[can use](#)

[Configuring the command](#)

[prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command](#)

[line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the](#)

[command shell](#)

[Tskill and Taskkill](#)



[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

Service Installer (InstSrv) is a command-line tool that installs and uninstalls executable services and assigns names to them.

Kernrate.exe: Kernel Profiling Tool

Kernel Profiling Tool (KernRate) is a command-line tool that is a CPU sample profiler. A sample profiler monitors performance and reports back to the user. KernRate reports on kernel and user-mode processes to provide information about CPU activity. Use KernRate to identify which processes are causing a CPU bottleneck.

Linkspeed.exe: Link Speed

Link Speed is a command-line tool that displays the speed of the connection between computers on a network.

Memmonitor.exe: Memory Monitor

MemMonitor is a command-line tool that monitors the memory a process uses

Memtrriage.exe: Resource Leak Triage Tool

MemTriage is a command-line tool that detects a possible resource leak on a running system. MemTriage records process information or current kernel pool information and saves it to a log file.

Now.exe: STDOUT Current Date and Time

Now reads standard input (STDIN) and then displays, on standard output (STDOUT), the current date and time followed by the STDIN. Used alone, it displays the current date and time.

Ntimer.exe: Windows Program Timer

NTimer is a command-line tool that measures how long a program runs. NTimer shows elapsed time, time in user mode, and time in privileged mode.

Oh.exe: Open Handles

Open Handles is a command-line tool that shows the handles of all open windows. OH can also be used to show only information about a specific process, object type, or object name.

Oleview.exe: OLE/COM Object Viewer

OLE/COM Object Viewer (OLEView) is a GUI tool that allows you to manage all Microsoft Component Object Model (COM) classes installed on your computer.

Pathman.exe: Path Manager

PathMan is a command-line tool that adds or removes components from system or user paths.

Pmon.exe: Process Resource Monitor

Process Resource Monitor (PMon) is a command-line tool that displays several measures of the CPU and memory use of processes running on the system. The PMon display appears in the command window.

Printdriverinfo.exe: Drivers Source

Drivers Source (PrintDriverInfo) is a command-line tool that is used to collect information about printer drivers and is primarily used for support purposes.

Qgrep.exe

Qgrep is a command-line tool that is used to search a file or list of files for a specific string or pattern and return the line containing the match. QGrep also allows you to search multiple files and subdirectories. Qgrep is similar to the UNIX tool "grep".

Remapkey.exe: Remap Windows Keyboard Layout

RemapKey is a GUI tool that changes the layout of a keyboard by remapping the scan codes of the keys.

Robocopy.exe: Robust File Copy Utility

A versatile utility for maintaining an identical copy of a folder and its sub-folders in more than one location,

Setprinter.exe: Spooler Configuration Tool

Spooler Configuration Tool is a command-line tool that is used to set configurations of local and remote printers

Sleep.exe: Batch File Wait

Sleep is a command-line tool that causes the computer to wait for a specified amount of time. For use in batch files

Splinfo.exe: Print Spooler Information

---

SplInfo is a command-line tool that collects information from the print spooler and displays it.

Tail.exe

Tail is a command-line tool that displays a user-specified number of the last lines of a text file, such as a log file, in a console window.

[Back to top](#)

# The Command Line in Windows

## Managing Windows XP Services with the Service Controller Command SC

### Site navigation

[Home](#)
[Assoc and Ftype](#)
[Batch files- basics](#)
[Batch files - branching](#)
[Batch files- iterating](#)
[Command Line-](#)
[Introduction](#)
[Command line list and reference](#)
[Commands that everybody can use](#)
[Configuring the command prompt window](#)
[Doskey](#)
[File system utility- Fsutil](#)
[Net Services](#)
[Netstat](#)
[Network Services Shell](#)
[PowerShell](#)
[Recovery Console](#)
[Recovery Console- Commands](#)
[Registry editor console](#)
[Scripts in the command line](#)
[Server 2003 tools for XP](#)
[Service Controller](#)
[Command \(SC\)](#)
[Shell command](#)
[Start-Run line](#)
[Support tools](#)
[Tasklist](#)
[TCP/IP networking tools](#)
[Tips for using the](#)
[command shell](#)
[Tskill and Taskkill](#)

The Service Controller utility SC is a powerful command-line utility for managing Windows services. Its various capabilities and functions are discussed here.

Many processes and functions of the Windows XP operating system and other software are classified under the general rubric of "services". Managing services with the graphical facility called the Services Console is [discussed at a sister site](#). In addition to a GUI method of managing services, Windows XP also has a powerful command-line utility. This utility, the Service Controller, is opened by entering "sc" into the command prompt and contains a large assembly of subcommands that we will survey.

The command-line method of managing services has the advantage of being available for scripts. It also allows for quickly stopping and starting services for troubleshooting purposes. Systems administrators use it for managing services on networks and for very detailed configuration. For the average PC user, it provides a quick and easy way to turn services on and off to see how system performance is affected.

### The SC subcommands

The "sc" command comes with numerous subcommands. A list can be seen at the this [Microsoft page](#) or by entering "sc /?" into a command prompt. There is also a list in the Windows XP *Help and Support Center*. Altogether, 24 subcommands are listed. Each subcommand in turn may have a subset of different commands. The table below shows a selection of the subcommands and their functions that are of most relevance to a typical PC owner. Much more detailed information is available at the XP *Help and Support Center* by searching "sc".

Table I. Selected subcommands for SC

Command	Function
sc config	Configures service startup and login accounts
sc continue	Resumes a paused service
sc enumdepend	Lists the services that cannot run unless the specified service is running
sc failure	Specifies what action to take upon failure of the service
sc pause	Pauses a service
sc qc	Displays the configuration of a particular service
sc query	Displays information about the specified service, driver, type of service, or type of driver
sc start	Starts a service running
sc stop	Sends a STOP request to a service (not all will respond)

### Examples of some useful ways to apply SC

The suite of commands that are available are very powerful and allow for much configuring of services. Although not all

[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

functions will be of interest to the average PC user, some are applicable to everyday experience. You can learn if a service is running, stop, start, or pause it, and determine if it will run when the system is started up. Here are examples of some commands that I think might be of interest.

### sc config

This command has a number of functions but one is to determine the status of a service at system startup. A service can be set to run automatically, manually or not at all. The commands are

```
sc config ServiceName start= flag
```

Here *ServiceName* is the name of the service and *flag* has one of the values *auto*, *demand*, or *disabled*. For example, to set a service to run manually the command is

```
sc config ServiceName start= demand
```

Note that there must be a space after the equals sign. The correct value for the parameter *ServiceName* may not always be obvious and the next command can be used to find it for all services.

### sc query

Information about services and drivers can be obtained with this command. Used alone it returns a list of running services with various information about the service. Lists can be inconvenient to read on a screen and they can be redirected to a text file. To create a text list of running services use the command

```
sc query > serviceslist.txt
```

The path for the text file *serviceslist.txt* can be anywhere that is convenient. To create a list of all services, use

```
sc query type= service state= all > allserviceslist.txt
```

To create a list of active drivers, use

```
sc query type= driver
```

Or for a list of everything, use

```
sc query state= all
```

### sc start

To start up a service that is not running, use

```
sc start ServiceName
```

### sc stop

To stop a running service, use

```
sc stop ServiceName
```

However, some services cannot or should not be stopped.

[Back to top](#)

# The Command Line in Windows

## Accessing System Folders with the Shell Command in Windows Vista

A number of system folders in Vista are most easily accessed with the "Shell:" command. The folders are listed.

Sometimes it's desired to work with one of the special system folders. As [discussed on another page](#), Windows XP system folders can be opened by entering the folder name in the Run line. However, in Vista this procedure does not work. Instead, the folder name has to be preceded with the command "shell:". Although the Run line can still be used, it is not shown in the default Vista configuration and the Start menu's "Start Search" line can be used instead. For example, to open the "SendTo" folder, enter

```
shell:sendto
```

Note that there must be no spaces between "shell:" and the command. Also note the colon. The command is not case-sensitive.

The command prompt can also be used to open special system folders with the shell command. In the command prompt the shell command must be used in conjunction with the "start" command. For example, to open the "Cookies" folder the command would be:

```
start shell:cookies
```

This method of opening system folders is also applicable in Windows XP.

Below is a list of system folders that can be opened. The list is taken [from the Registry key](#)

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\explorer\FolderDescriptions
```

(The list will vary somewhat depending on the version of Vista.)

### Vista System Folders

AddNewProgramsFolder	MyComputerFolder
Administrative Tools	NetHood
AppData	NetworkPlacesFolder
AppUpdatesFolder	OEM Links
Cache	Original Images
CD Burning	Personal
ChangeRemoveProgramsFolder	PhotoAlbums
Common Administrative Tools	Playlists
Common AppData	PrintersFolder
Common Desktop	PrintHood
Common Documents	Profile
Common Programs	ProgramFiles
Common Start Menu	ProgramFilesX64 (in 64-bit systems)
Common Startup	ProgramFilesX86 (in 64-bit systems)
Common Templates	ProgramFilesCommon
CommonDownloads	ProgramFilesCommonX64 (in 64-bit systems)
CommonMusic	ProgramFilesCommonX86 (in 64-bit systems)

### Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the](#)

[command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

CommonPictures

CommonVideo

ConflictFolder

ConnectionsFolder

Contacts

ControlPanelFolder

Cookies

CredentialManager

CryptoKeys

CSCFolder

Default Gadgets

Desktop

Downloads

DpapiKeys

Favorites

Fonts

Gadgets

Games

GameTasks

History

InternetFolder

Links

Local AppData

LocalAppDataLow

LocalizedResourcesDir

MAPIFolder

My Music

My Pictures

My Video

Programs

Public

PublicGameTasks

Quick Launch

Recent

RecycleBinFolder

ResourceDir

SampleMusic

SamplePictures

SamplePlaylists

SampleVideos

SavedGames

Searches

SearchHomeFolder

SendTo

Start Menu

Startup

SyncCenterFolder

SyncResultsFolder

SyncSetupFolder

System

SystemCertificates

SystemX86

Templates

TreePropertiesFolder

UserProfiles

UsersFilesFolder

Windows

[Back to top](#)

# The Command Line in Windows

## Site navigation

powered by [FreeFind](#)

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

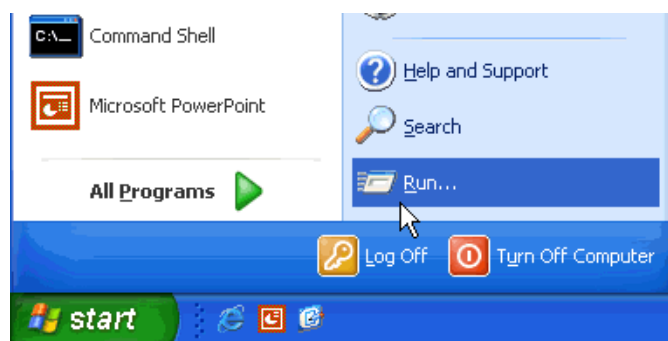
[Tips for using the](#)

## The Start-Run Line

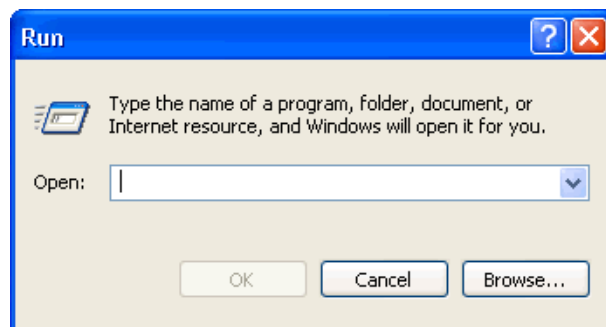
The "Run" line in the Start menu can be used to speed up access to a whole variety of functions. Some examples of the shortcuts that are available are discussed here.

### Introduction to the Run Line

The Run command line may be one of the least utilized functions in the Start menu. This is a pity since it can be very useful. It is often the quickest way to launch programs or to open folders and documents. The figure below shows the Start-Run entry.



Clicking the entry "Run" opens the box shown below, where commands may be typed and entered.



### Opening applications in Run

Although applications can be opened in a variety of ways, the Run line often provides the quickest route. Desktop shortcut icons are also a quick route but you have to know how to create a shortcut for all the applications and you may end up with dozens of icons.

The best candidates for the Run line are applications that are in the "path" environment. (Go to [this page](#) for more discussion of the path.) The path environment is a set of folders whose names do not have to be included when entering a command. The path environment variable normally includes `Windows\` and `Windows\system32\`. Many common accessories and Windows applets are in these folders and can be opened by entering just the executable file name. Several that I use constantly are the Registry editor (`regedit`) and the [System Configuration Utility](#) (`msconfig`). Note that neither of these frequently used system tools has an entry in Start- All Programs. The Run line is the standard method of accessing them.

A table listing some applications that can be opened in the Run line is given below.

*A few applications for the Run line*



[command shell](#)[Tskill and Taskkill](#)[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)

Entry for Run	Function
calc	Opens calculator
cmd	Opens command prompt window
explorer	Opens Windows explorer
magnify	Screen magnifier accessory
msconfig	System Configuration Utility
mshearts	Opens Hearts game
msinfo32	System Information applet
mspaint	Opens graphics accessory Paint
notepad	Notepad accessory
regedit	Registry editor
sol	Opens Classical Solitaire game

Programs not in the path require their full address, including the root folder and all subfolders. Actually typing long path names is not required since a "Browse" function comes with the Run line. This provides a way to search for files of interest and to enter them directly without typing. If desired, frequently used programs can be added to the path environment using the methods that are discussed in a [section below](#).

## Opening Management Consoles

Some of the functions that I find convenient to open by means of the Run line include various [Management Consoles](#). For example, the [disk defragmenter](#) is opened by entering "dfrg.msc". I find this easier than the multi-step process involved otherwise. A list of the appropriate file names for opening some of the various services is given in the table below. A more complete list [is on this page](#).

*Commands for some Management Consoles (msc extension required)*

Entry for Run	Function
<i>ciadv.msc</i>	Manages the Indexing Service
<i>compmgmt.msc</i>	Computer Management Console. It contains a number of the other consoles
<i>devmgmt.msc</i>	Device Manager
<i>dfrg.msc</i>	Disk Defragmenter
<i>diskmgmt.msc</i>	Disk Management
<i>gpedit.msc</i>	Group Policy Editor. Windows XP Professional only
<i>services.msc</i>	Manages the many services involved in Windows and installed software

## Opening Control Panel Applets

It is also possible to use Run to open the applets that appear in the *Control Panel*. A full discussion of shortcuts to Control

Panel applets is given [on this page](#). For example, entering "main.cpl" launches the mouse properties window.

## Rundll32.exe

There are a number of commands employing Rundll32.exe that can be entered into Start-Run. A full discussion can be [found here](#).

## Opening folders in Run

Not only files but also folders can be opened in the Run line. Folders contained by a folder in the path are in this category. Examples are folders within `Windows\` and `Windows\system32\` such as the folders [Fonts](#) and "Drivers". Folders that are in `\Documents and Settings\{Current User}\` can also be opened in Run. An example is [SendTo](#) (written as one word). Being able to open this folder in Run is convenient for editing. It makes it easier to add functions to the "Send To" entry in the right-click context menu. ([See this page](#)) Another example of a folder from the same location that can be entered is *Cookies*.

*Note about Vista: Certain system folders like SendTo and Cookies are not directly accessible in Vista. See the [page on the Shell command](#).*

There are also some interesting shortcuts to folders that are available in Run. Typing the backslash (\) in the run line and entering it brings up the root folder, usually the C: drive. Typing and entering a period (.) brings up the folder `\Documents and Settings\{Current User}\` in Windows XP (or `Users\{Current User}` in Vista). Entering two periods (..) opens the folder `\Documents and Settings\` (or `Users` in Vista).

## Dragging and dropping folders and files into the Run line

If the Run line is open (make sure it is empty) folders or files can be dragged and dropped on it from an open folder window. The full path of the dropped object will be inserted into the Run line and clicking "OK" or pressing the "Enter" key will open the dropped file or folder. Although this feature presents no particular advantage in general, it can be helpful to those who have trouble with double-clicking the mouse.

## Adding applications to the Path

The ability to enter a short file name into the Run line to open a program can be extended to any program by putting the folder containing the program executable into the path. Adding folders to the path is [described here](#).

Alternatively, the Registry can be edited to explicitly contain the path to the desired executable file or files. The Registry key involved is

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\App Paths
```

1. Create a new sub-key with the name of the executable file that you wish to add to the path. e.g.,

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\App Paths\somefile.exe
```
2. In this new key, add a string variable named "Path" containing the value of the the path to your new executable file, e.g.,

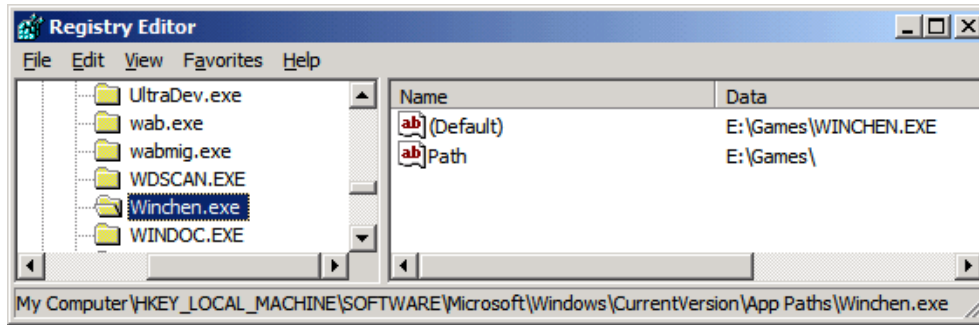
```
C:\Program files\newprogramfolder\
```
3. The new key will already have an empty variable (Default). Edit it to have the string value of entire address of the new program executable , e.g.,

```
C:\Program files\newprogramfolder\somefile.exe
```

You can now enter "somefile.exe" into the Run line to open the program. An example of what the Regedit entries look like is shown in the figure below. I have added a chess game with the executable "winchen.exe" to the path.

A number of files are likely to have already been added on your system. Many applications place themselves here when they are installed. Examples are Microsoft Office components. It is sufficient to enter "winword" into the Run line to open Microsoft

Word or "excel" to open Microsoft Excel.



## Internet Applications

If you are connected to the Internet, entering an URL into Run opens Internet Explorer (or other browser if it is properly associated) and takes you right to the appropriate web site. The "Browse" function can be used to go to your Favorites folder and you can then click on a link. (Be sure the Browse function is showing "All files" as the file type.) On some systems it will even initiate a Web connection if you are not already on-line. You can also start e-mail by entering "mailto:someone@somewhere.com". This will open a blank new e-mail with the address already entered. If you want to use an e-mail client other than the Microsoft application Outlook Express, it will have to be associated with the "mailto" function. Many e-mail clients do this automatically when they are installed. (As far as I know, this does not apply to AOL.)

Google supports a command line function that allows for simple searches on a single term. Enter "www.google.com/search?q=%1" into the Run line, where %1 is the term that is being searched. A dash can be used to combine words. For example, try "www.google.com/search?q=windows-registry" (Omit the quotes.)

## Comparison of Run with the Command Prompt

Although a great many commands can be executed in either the Run line or a command prompt, some commands will run directly only in one or the other. Commands which are built into the command interpreter cannot be entered in the Run line without first invoking *cmd.exe*. They are [listed here](#). These include commands like "dir" and "del". Certain special features of the Run line such as the direct way of opening folders or the Internet shortcuts discussed above do not work in a command prompt unless prefaced with the [command "Start"](#).

## Accessing the Command Shell from Run

The command interpreter can be invoked to carry out a command from the Run line by entering

```
cmd /c some_command
```

With the switch "/c", *some\_command* will be carried out and the command shell will then close. If you want the command shell to remain open, use the switch "/k". Enter

```
cmd /k some_command
```

## The Run Line in Vista

The *Start* menu in Vista has no *Run* line in its default setting. Many of the functions of *Run* can be carried out in the new Search function that is at the bottom of the Vista *Start* menu but I still like to use *Run* sometimes. You can get *Run* back temporarily by using the keyboard shortcut **Windows key+R**. To put *Run* permanently back in the *Start* menu :

1. Right-click on the *Start* menu and choose "Properties"
2. Select the "Start Menu" tab and click on the "Customize..." button
3. Check the "Run command" option

[Back to top](#)



# The Command Line in Windows

## Site navigation

[Home](#)
[Assoc and Ftype](#)
[Batch files- basics](#)
[Batch files - branching](#)
[Batch files- iterating](#)
[Command Line-](#)
[Introduction](#)
[Command line list and](#)
[reference](#)
[Commands that everybody](#)
[can use](#)
[Configuring the command](#)
[prompt window](#)
[Doskey](#)
[File system utility- Fsutil](#)
[Net Services](#)
[Netstat](#)
[Network Services Shell](#)
[PowerShell](#)
[Recovery Console](#)
[Recovery Console-](#)
[Commands](#)
[Registry editor console](#)
[Scripts in the command](#)
[line](#)
[Server 2003 tools for XP](#)
[Service Controller](#)
[Command \(SC\)](#)
[Shell command](#)
[Start-Run line](#)
[Support tools](#)
[Tasklist](#)
[TCP/IP networking tools](#)
[Tips for using the](#)
[command shell](#)
[Tskill and Taskkill](#)

## Xcopy and its Application

The syntax and use of the command "Xcopy" is described.

Of all the command line executables, Xcopy is one of the most useful for the average home PC user. It provides a powerful and versatile method for copying and backing up files and directories.

### Syntax of Xcopy command

Xcopy has a large number of possible switches, which gives the command a great deal of flexibility. The syntax for the command is given by:

```
XCOPY source [destination] [/A | /M] [/D[:date]] [/P] [/S [/E]] [/V] [/W] [/C] [/I] [/Q]
[/F] [/L] [/G] [/H] [/R] [/T] [/U] [/K] [/N] [/O] [/X] [/Y] [-Y] [/Z]
[/EXCLUDE:file1[+file2][+file3]...]
```

Upper case letters have been used above but the command is case-insensitive. A description of the various switches is given in Table I. Note that Windows Vista has an additional switch "/B". The function of this switch is to copy a Symbolic Link itself instead of the target of the link.

Table I. Description of switches for the command Xcopy

Switch	Description
/A	Copies only files with the archive attribute set, doesn't change the attribute.
/M	Copies only files with the archive attribute set, turns off the archive attribute. Useful in backup.
/D:m-d-y	Copies files changed on or after the specified date. If no date is given, copies only those files whose source time is newer than the destination time. Useful in backup.
/P	Prompts you before creating each destination file.
/S	Copies directories and subdirectories except empty ones.
/E	Copies directories and subdirectories, including empty ones. Same as /S /E. May be used to modify /T.
/V	Verifies each new file. Not used by Windows XP.
/W	Prompts you to press a key before copying.
/C	Continues copying even if errors occur.
/I	If destination does not exist and copying more than one file, assumes that destination must be a directory.

[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

/Q	Does not display file names while copying.
/F	Displays full source and destination file names while copying.
/L	Displays files that would be copied.
/G	Allows the copying of encrypted files to destination that does not support encryption.
/H	Copies hidden and system files also.
/R	Overwrites read-only files.
/T	Creates directory structure, but does not copy files. Does not include empty directories or subdirectories.
/U	Copies only files that already exist in destination.
/K	Copies attributes. Normal Xcopy will reset read-only attributes.
/N	Copies using the generated "short" names. May be necessary when copying from NTFS to FAT16.
/O	Copies file ownership and ACL information.
/X	Copies file audit settings (implies /O).
/Y	Suppresses prompting to confirm that you want to overwrite an existing destination file. May be preset in the COPYCMD environment variable
/-Y	Prompts to confirm you want to overwrite an existing destination file.
/Z	Copies over a network in restartable mode.
/EXCLUDE:file1[+file2][+file3]...	Specifies a list of files containing strings to be excluded. Tricky to use. See below.

## Applications of Xcopy

Xcopy provides an excellent tool for backing up selected folders. With appropriate switches, a variety of backup scenarios can be created. One possible backup configuration would be to copy only those files that have been changed. Here is an example command:

```
xcopy C:\somefolder E:\backupfolder /D /E /C /R /H /I /K /Y
```

This command will copy all files, including those in sub-folders, that are newer in the source folder. It will copy hidden as well as read-only files and will create the destination folder and/or sub-folders if they do not already exist.

The next example shows the use of a wildcard; it collects all files of a given type and copies them into one place. The command

```
xcopy C:\*.mp3 E:\mp3folder /S /I /C > E:\mp3List.txt
```

will collect all MP3 files on the C: drive and copy them to a folder on the E: drive. It also creates a list of the files copied and places the list on the E: drive. This simple command preserves the sub-folder structure. If desired, a more advanced script could be written that places all MP3 files in one folder with no sub-folders.

## Excluding files and folders

A useful feature of Xcopy is the ability to exclude certain files and/or folders from being copied by means of the "/Exclude" switch. All names containing a given string of characters can be excluded. Unfortunately, [Microsoft's description](#) of the switch is

neither a model of clarity nor accurate. In fact, postings on the Internet indicate considerable confusion exists about how to implement this handy feature. Perhaps the list below will help clarify how to use it.

1. The strings contained in the names of the files and/or folders to be excluded are not entered directly in the command switch itself.
2. The entry in the command switch is one or more text files that list the strings to be excluded.
3. The format of the switch is

```
/Exclude:{path}list_of_exclusions.txt
```

Note the colon between Exclude and the name of the file that lists the exclusions. Although more than one exclusion file can be used, stick with one if you can.

4. The name of the file that lists exclusions cannot have spaces. Quoting does not help.
5. Be careful with pathnames. Remember that operations are relative to the working directory for the command prompt. I suggest placing the exclusion list in the top directory of the directories being copied and opening the command prompt in the directory just above the directory and sub-directories to be copied. ([Go here if you don't know how](#) to open the command prompt wherever you want.) The various relative paths will then be simple. If you open the command prompt in the default location, things can get complicated.
6. The structure of the exclusions list is one exclusion string per line. Wildcards are not used and do not work. If you want to exclude all files whose names contain ABC, simply place ABC on one line of the exclusion file. If it is being used together with the switch "/S", the exclusion file will apply to sub-directories as well as the main directory.
7. To exclude a particular sub-directory, place its name in the exclusion file with backward slashes before and after its name, viz.,

```
\excluded_directory\
```

8. Use a text editor to create exclusion files. Do not use Microsoft Word or other word processor unless you are careful to create text files. Use ANSI encoding. When I tried UTF-8 encoding, the file did not work.

## Exit codes for Xcopy

Many commands issue integer exit codes to indicate the status of the command after it is run. The exit codes for Xcopy are given in Table II. When placed in ["if" statements](#), these error codes can be useful in batch files .

*Table II. Exit codes for Xcopy*

Exit code	Description
0	Files were copied without error.
1	No files were found to copy.
2	The user pressed Ctrl+C to terminate xcopy.
4	Various errors including insufficient memory or disk space, an invalid drive name, or invalid syntax.
5	Disk write error occurred.

[Back to top](#)



# The Command Line in Windows

## Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the](#)

[command shell](#)

[Tskill and Taskkill](#)

## TCP/IP and Networking Tools

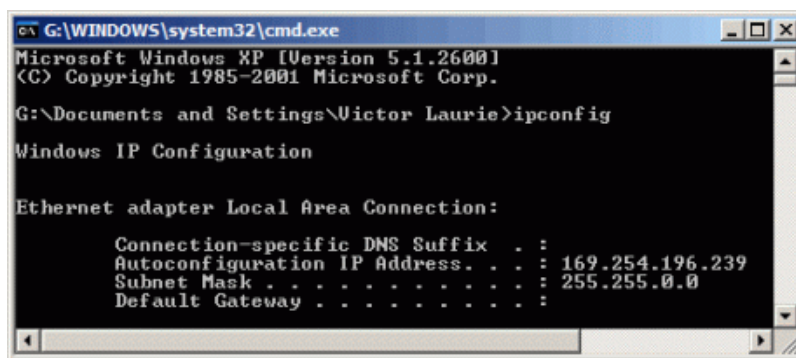
Windows XP has a whole array of helpful command line tools for configuring and testing Internet and LAN connections. On this page is a discussion of some of the networking tools that can be useful to an average PC user.

There are dozens of networking tools available for Windows XP (For example, see this [command-line reference](#).) Most of these are specialized and are mainly of interest to professionals who are maintaining a large network. Many, however, are relevant to the Internet and some of these can be helpful to the average PC user. The tools of interest to the discussion here are a few of the TCP/IP utilities. (Go [here](#) for a complete list.) [TCP/IP](#) refers to the set of protocols that are used for Internet connections and on most networks. Discussing TCP/IP is beyond the scope of this page but more details are available [on a separate page](#) and in the references in the sidebar. Fortunately, it is not necessary to understand the gory details of TCP/IP in order to make practical use of the tools considered here.

All of the tools are run by opening a Command window and entering the appropriate command. Go to Start-Run and enter "cmd" to open a Command window.

### Windows IP Configuration Tool (ipconfig)

The Windows IP Configuration tool (*ipconfig*) is the command-line equivalent of the accessory "Winipcfg" that was present in Windows 9X/Me. It is used to display the TCP/IP network configuration values. To open it, enter "ipconfig" in the command prompt. If you are connected directly to the Internet, you will obtain your IP address. (For a discussion of what an IP is, [go here](#).) The figure below shows the result for a broadband connection where the IP is assigned automatically. Here the IP is your computer's temporary address on the Internet.



```
G:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

G:\Documents and Settings\Victor Laurie>ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix . . . : 
    Autoconfiguration IP Address. . . . : 169.254.196.239
    Subnet Mask . . . . . : 255.255.0.0
    Default Gateway . . . . . :
```

If you are on a local area network using a router, the information is different. You do not obtain the IP corresponding to the network's address on the Internet. (To obtain the IP that the Internet sees, go to a source such as [DSL Reports Whois](#).) The IP given is the *local* address on the network. This information can be helpful in diagnosing network problems. Also listed is the "gateway" or router address on the local network. The figure below illustrates the result.

[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

```
G:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

G:\Documents and Settings\Victor Laurie>ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : 
    IP Address . . . . . : 192.168.1.100
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.1.1

G:\Documents and Settings\Victor Laurie>
```

## Switches for IPConfig

There are also a variety of switches for *ipconfig* that add functions. These are invoked by entering "*ipconfig /{switch}*". To obtain a list of switches, enter "*ipconfig /?*" or "*ipconfig -?*". These are shown in the figure below. The switches of most interest to everyday use are "*release*" and "*renew*". Note that IP addresses are typically assigned or "leased" for a period of time, often a day or more. It sometimes happens that IP addresses are no longer valid or are in conflict. Problems can often be solved by first releasing the IP address and then renewing it. Sometimes cable or DSL modems that seem to be disabled can be restored this way. If you travel and use broadband connections elsewhere, you will often find this procedure of releasing and renewing the IP address to be necessary.

```
C:\>ipconfig /?

USAGE:
    ipconfig [/? | /all | /renew [adapter] | /release [adapter] |
           /flushdns | /displaydns | /registerdns |
           /showclassid adapter |
           /setclassid adapter [classid] ]

where
    adapter      Connection name
                  (wildcard characters * and ? allowed, see examples)

Options:
    /?           Display this help message
    /all         Display full configuration information.
    /release     Release the IP address for the specified adapter.
    /renew       Renew the IP address for the specified adapter.
    /flushdns    Purges the DNS Resolver cache.
    /registerdns Refreshes all DHCP leases and re-registers DNS names
    /displaydns  Display the contents of the DNS Resolver Cache.
    /showclassid Displays all the dhcp class IDs allowed for adapter.
    /setclassid  Modifies the dhcp class id.
```

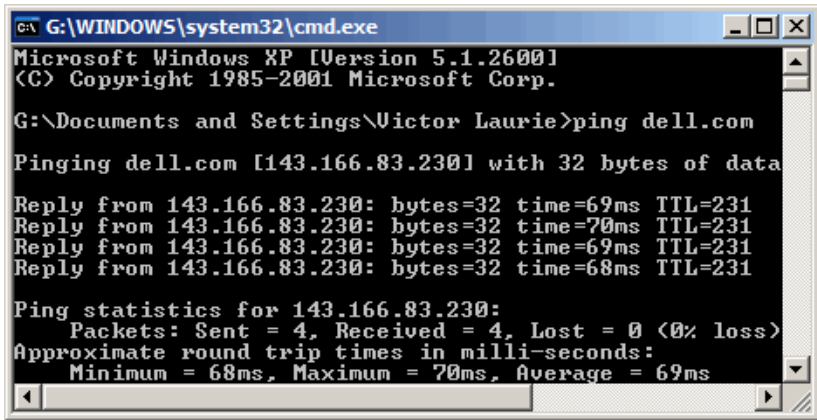
For a detailed output of network parameters, you can use the command "*ipconfig /all*". Unless you are experienced with networks, however, this may be more than you want to know.

The switches "*flushdns*" and "*displaydns*" are also sometimes useful in everyday use and they are discussed on [another page](#) at a sister site.

## Ping

*Ping* is an old Unix tool that has been around for a long time but many PC users are unfamiliar with the Windows version. *Ping* sends out a packet to a designated internet host or network computer and measures its response time. The target computer will return (hopefully) a signal. It is a way of determining the quality of your connection to another site. You will also receive an IP address that corresponds to the user-friendly type of URL (see [this page](#) for further discussion of IPs and URLs). To use ping, open a command window (or DOS in Windows 9X/Me) and type: `ping <hostname>`. For example, to ping Dell enter: `ping dell.com` Please note the use of a hostname, not a complete URL. The prefix "http://" is never used. Many sites also do not

require "www" . By convention, 32 byte packets will be transmitted four times. You will receive a screen output that looks like:



```
G:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

G:\Documents and Settings\Victor Laurie>ping dell.com

Pinging dell.com [143.166.83.230] with 32 bytes of data:

Reply from 143.166.83.230: bytes=32 time=69ms TTL=231
Reply from 143.166.83.230: bytes=32 time=70ms TTL=231
Reply from 143.166.83.230: bytes=32 time=69ms TTL=231
Reply from 143.166.83.230: bytes=32 time=68ms TTL=231

Ping statistics for 143.166.83.230:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss)
    Approximate round trip times in milli-seconds:
        Minimum = 68ms, Maximum = 70ms, Average = 69ms
```

The screen tells me that the IP for *dell.com* is 143.166.83.230 (For the moment, at least. This can change.) It also tells me that the average round trip time for a packet was 69 milliseconds, which means I have a good connection to *dell.com*. Long reply times of several hundred milliseconds are indicative of a slow connection. Note that some major sites such as *microsoft.com* do not like being pinged and block pings. In that case you will get a "Request timed out" message.

In addition to being used on the Internet, Ping is often used to test connections on local networks. More details can be found in this [Microsoft article](#).

## Tracert

*Tracert* (traceroute) is another old tool borrowed from Unix. The actual path between two computers on the Internet is not a straight line but consists of numerous segments or "hops" from one intermediate computer to another. *Tracert* shows each step of the path taken. It can be interesting to see just how convoluted it is. The times for each hop and the IP addresses for each intermediate computer are displayed. *Tracert* shows up to 30 hops. It is convenient for finding if there is one particular segment that is causing a slow or bad connection. A typical command might be "tracert dell.com".

## Pathping

This command combines functions of *Ping* and *Tracert*. *Pathping* will first list the number of hops required to reach the address you are testing and then send multiple pings to each router between you and the destination. After that, it computes results based on the packets returned from each router. Because pathping displays the degree of packet loss at any given router or link, you can determine which routers or subnets might be having network problems. Note that the whole process may consume 5-10 minutes because many pings are being sent. There are switches to modify the process and these can be seen by entering "pathping /?" in the command prompt.

## Netstat

*Netstat* displays the active TCP connections and ports on which the computer is listening, Ethernet statistics, the IP routing table, statistics for the IP, ICMP, TCP, and UDP protocols. It comes with a number of switches for displaying a variety of properties of the network and TCP connections. (*One tricky point: the switches must be prefixed with a minus, not a slash.*) More detail is [at this page](#). One possible use for *Netstat* is to determine if spyware or Trojans have established connections that you do not know about. The command "netstat -a" will display all your connections. The command "netstat -b" will show the executable files involved in creating a connection. A figure showing all the switches and syntax is given below.

```
Displays protocol statistics and current TCP/IP network connections.

NETSTAT [-a] [-b] [-e] [-n] [-o] [-p proto] [-r] [-s] [-v] [interval]

-a           Displays all connections and listening ports.
-b           Displays the executable involved in creating each connection or
            listening port. In some cases well-known executables host
            multiple independent components, and in these cases the
            sequence of components involved in creating the connection
            or listening port is displayed. In this case the executable
            name is in [] at the bottom, on top is the component it called,
            and so forth until TCP/IP was reached. Note that this option
            can be time-consuming and will fail unless you have sufficient
            permissions.
-e           Displays Ethernet statistics. This may be combined with the -s
            option.
-n           Displays addresses and port numbers in numerical form.
-o           Displays the owning process ID associated with each connection.
-p proto     Shows connections for the protocol specified by proto; proto
            may be any of: TCP, UDP, TCPv6, or UDPv6. If used with the -s
            option to display per-protocol statistics, proto may be any of:
            IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, or UDPv6.
-r           Displays the routing table.
-s           Displays per-protocol statistics. By default, statistics are
            shown for IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, and UDPv6;
            the -p option may be used to specify a subset of the default.
-v           When used in conjunction with -b, will display sequence of
            components involved in creating the connection or listening
            port for all executables.
interval    Redisplays selected statistics, pausing interval seconds
            between each display. Press CTRL+C to stop redisplaying
            statistics. If omitted, netstat will print the current
            configuration information once.
```

## Nslookup

This command helps diagnose the Domain Name System (DNS) infrastructure and comes with a number of sub-commands. These are mainly for systems administrators. The primary interest for average PC users is its use to find the computer name corresponding to a numeric IP. For example, if you want to know who is "216.109.112.135", enter "nslookup 216.109.112.135" and you will find that it is (or was anyway) a Yahoo computer. My firewall keeps a log of the IPs involved in the attempts to probe my computer and I sometimes look a few up to see who they are. (There are also *Whois* search sites available on the Web as mentioned in the *Ipconfig* section.)

## Netsh

The network services shell is a large suite of many tools. I discuss it in some depth on [another page](#).

[Back to top](#)

# The Command Line in Windows

## Windows XP SP2 Support Tools

Windows systems come with numerous command-line tools. There are also others that can be downloaded and installed. One group is called support tools and a number of these are listed here. The tools are primarily for system administration but some can also be applied to diagnosing and resolving computer problems in PCs in the home.

The list below omits some of the more specialized tools, such as those for Active Directory. A [complete list is here](#) along with the free download. These tools are in addition to the large number of command-line utilities that are part of the default installation of Windows XP Professional. These additional tools require their own installation . (Not available for 64-bit Windows XP.)

### A Selection of Windows XP Command-Line Support Tools

- *acldiag.exe*- manages access control lists
- *activate.exe*- Windows product activation
- *apmstat.exe*- provides status information on Advanced Power Management (APM) features.
- *bindiff.exe*- shows the differences between two binary files
- *bitsadmin.exe*- manages the Background Intelligent Transfer Service
- *browstat.exe*- a general purpose character-based browser diagnostic tool
- *cabarc.exe*- allows users to create, query and extract Windows cabinet (CAB) files.
- *depends.exe*- provides way to determine which DLLs an application depends on ( also known as "Dependency Walker")
- *dhcplloc.exe*- displays the DHCP servers active on your subnet
- *diruse.exe*- displays directory size information for NTFS volumes
- *dmdiaq.exe*- displays system state and configuration information describing disk storage.
- *dupfinder.exe*- duplicate file finder
- *efsinfo.exe*- displays information about files that are encrypted with Encrypting File System (EFS) on NTFS partitions.
- *extract.exe*- a utility that allows you to extract all files or specific files contained within a cabinet (.cab) file
- *filever.exe*- displays information on the versions of executable files
- *ipseccmd.exe*- configures Internet Protocol Security (IPSec) policies
- *memsnap.exe*- takes a snapshot of the memory resources being consumed by all running processes and writes this information to a log file
- *msicuu.exe*- Windows Installer Clean Up Utility
- *msizap.exe*- removes either all Windows Installer information for a product or all products installed on a computer
- *netcap.exe*- monitors packets on a LAN and writes the information to a log file
- *netdiag.exe*- tests the network connectivity
- *netset.exe*- used to add, remove, or change the network configuration
- *pfmon.exe*- displays the faults that occur while executing a process
- *pstat.exe*- gives you information about the processes and drivers that are currently running on your computer.
- *pviewer.exe*- process viewer
- *setx.exe*- sets environment variables
- *showaccs.exe*- enables users to examine the access control lists (ACLs)

### Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console- Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

- *timezone.exe*- Daylight Saving Time Update Utility
- *whoami.exe*- displays the user name and security identifier (SID)
- *windiff.exe*- shows the differences between specified ASCII text files, or folders containing ASCII text files
- *xcacs.exe*- used to set all file-system security options that are accessible in Windows Explorer from the command line

[Back to top](#)



# The Command Line in Windows

## Site navigation

[Home](#)
[Assoc and Ftype](#)
[Batch files- basics](#)
[Batch files - branching](#)
[Batch files- iterating](#)
[Command Line-](#)
[Introduction](#)
[Command line list and reference](#)
[Commands that everybody can use](#)
[Configuring the command prompt window](#)
[Doskey](#)
[File system utility- Fsutil](#)
[Net Services](#)
[Netstat](#)
[Network Services Shell](#)
[PowerShell](#)
[Recovery Console](#)
[Recovery Console-](#)
[Commands](#)
[Registry editor console](#)
[Scripts in the command line](#)
[Server 2003 tools for XP](#)
[Service Controller](#)
[Command \(SC\)](#)
[Shell command](#)
[Start-Run line](#)
[Support tools](#)
[Tasklist](#)
[TCP/IP networking tools](#)
[Tips for using the command shell](#)
[Tskill and Taskkill](#)

## Managing Windows XP Programs from the Command Line: Tasklist

Windows XP Professional comes with a powerful command-line tool called Tasklist that provides many details on the programs and processes that are running.

Many will be familiar with the graphical tool Task Manager, which I have [discussed elsewhere](#), and which provides various kinds of information about the applications and processes that are running on a system. There are also several command-line tools that provide similar but even more detailed information. In this article I will discuss the features of the tool called *Tasklist* (the system file is *tasklist.exe*). This tool is part of the regular installation of the Professional version of XP but does not come with the Home edition. However, those with the Home version of XP can [download Tasklist here](#). Tasklist can be applied to see how much memory and CPU time running processes are using, what DLL files they rely on, and other information. Thus it can be a very useful troubleshooting tool.

### Basic Tasklist command

If all you want to know is what tasks are running, enter TASKLIST into the command line. The output can be redirected to a file if you wish. The default format is a table with several columns of information. An example of a partial console output is shown in the figure below. There are five columns of information. The following list gives the meanings of the various column headings:

#### Image Name

The name of the process or the executable file running the process.

#### PID

The process ID. The system assigns a number to each process so it can keep track of it. It is possible to have several processes running with identical names but the PID will be unique. Note that the PID may not be the same each time you open a particular program. You may need the PID to run certain other diagnostic tools and Tasklist is one way to obtain this information.

#### Session Name

Unless you are on a network, this will read "Console" indicating that the process was started locally. Home PC users can usually ignore this column.

#### Session#

Each session is assigned a number. Home PC users can usually ignore this column also.

#### Mem Usage

This gives the very useful information about how much memory (in KB) that a process was using at the time Tasklist was run.



```
C:\>tasklist

Image Name                    PID Session Name        Session#    Mem Usage
=====
System Idle Process           0 Console              0            16 K
System                        4 Console              0           212 K
smss.exe                      520 Console             0           464 K
csrss.exe                     584 Console             0          1,160 K
winlogon.exe                  608 Console             0          4,732 K
services.exe                 652 Console             0          3,184 K
lsass.exe                    664 Console             0          1,380 K
svchost.exe                   824 Console             0          3,216 K
svchost.exe                   888 Console             0         18,020 K
svchost.exe                   988 Console             0          1,964 K
svchost.exe                  1016 Console            0          3,116 K
spoolsv.exe                  1048 Console             0          4,196 K
Explorer.EXE                 1596 Console             0          4,740 K
augasmvr.exe                 1692 Console             0          4,476 K
augpsvc.exe                  1772 Console             0          2,312 K
wdfmgr.exe                   1852 Console             0          1,564 K
usmon.exe                    1884 Console             0          9,052 K
OneTouch.exe                 112 Console             0          2,264 K
```

Additional columns will be displayed in the so-called "verbose" mode that is obtained with the switch `/v`. These columns are:

**Status**

Gives the current status of the process as "Running", "Not Responding", or "Unknown". Useful for finding hung processes. Unknown status may refer to a normal process but Not Responding indicates a process that should be stopped.

**User Name**

User account under which the process is running, Windows itself will be running many processes and the various system accounts SYSTEM, LOCAL SERVICE , or NETWORK SERVICE. will appear, coupled with the local domain name NT AUTHORITY.

**CPU Time**

The *total* amount of CPU cycle time used by the process since its start. This can be a big number if you never turn off the computer.

**Window Title**

Windows display name of the process if it exists. Can sometimes help identify what program is involved.

## More advanced options for Tasklist

There are many more options and these are provided by switches. The full syntax is:

```
TASKLIST [/S system [/U username [/P [password]]]] [/M [module] | /SVC | /V] [/FI filter]
[/FO format] [/NH]
```

Upper case has been used for clarity but the command is not case-sensitive. Table I describes the various parameters.

*Table I. Parameters for TASKLIST*

Parameter	Description
<code>/S system</code>	Specifies the remote system to connect to. Not needed for local computer
<code>/U username</code>	Specifies the user context. Not needed for local computer
<code>/P [password]</code>	Specifies the password for the given user context (if necessary).
<code>/M [module]</code>	Lists all tasks that have DLL modules loaded in them that match the given pattern name. If the module name is not specified, displays all modules loaded by each task.
<code>/SVC</code>	Displays services in each process.
<code>/V</code>	Specifies that the verbose information is to be displayed.
<code>/FI filter</code>	Displays a set of tasks that match a given criteria specified by the filter.

[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

### Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

<code>/FO format</code>	Specifies the output format. Valid values: "TABLE", "LIST", "CSV".
<code>/NH</code>	Specifies that the "Column Header" should not be displayed in the output. Valid only for "TABLE" and "CSV" formats.

These additional parameters enable Tasklist to provide very detailed information about the system. Some examples will be shown in the next sections.

### Find which Services use a process

It can be very useful to know the relationship between a process and the services that are running on a system (for a discussion of services [see this page.](#)) To obtain a table relating Image Name, PID, and Services use the command

```
tasklist /svc >list.txt
```

Here I have shown the redirect to a file to illustrate creating a text record. One application of this command is for diagnosing problems with a service by monitoring the memory usage and other properties of the processes associated with the service.

### Find which DLL files are used by a process

Processes can be using many different DLL files by calling on various procedures from their libraries. It is not uncommon for a problem to arise because a DLL is corrupted or is the wrong version. To find which DLLs are used by each process use the command

```
tasklist /m
```

This will return a table relating Image Name, PID, and Modules. "Modules" here indicates DLLs. The table may have quite a few entries and the list can be limited to a specific DLL by using its name in the command. For example, to see only the processes that use *oleaut32.dll*, enter

```
tasklist /m oleaut32.dll
```

### Filtering Tasklist output

The output can be narrowed down to specific parameters by using filters and the switch `/FI`. There are a number of comparison operators and these are given in Table II. Not all operators can be used with every parameter and allowed values are shown for the most useful parameters in Table III.

Table II. Comparison operators for filters

Operator	Description
eq	Equals
ne	Does not equal
gt	Greater than. Only used with numeric values
lt	Less than. Only used with numeric values
ge	Greater than or equal to. Only used with numeric values
le	Less than or equal to. Only used with numeric values

Table III. Filter operators and allowed values

Parameter	Valid operators	Valid values

ImageName	eq, ne	Any valid string
PID	eq, ne, gt, lt, ge, le	Any valid positive integer
MemUsage	eq, ne, gt, lt, ge, le	Any valid positive integer in kilobytes
Status	eq, ne	Running, Not Responding, Unknown
Username	eq, ne	Any valid user name (includes SYSTEM, LOCAL SERVICE , NETWORK SERVICE)
WindowTitle	eq, ne	Any valid string

An example of using a filter is a command to find processes that are not responding. The command would be

```
tasklist /fi "status eq not responding"
```

Another example is to find processes using a lot of memory, say more than 40 MB. The command is

```
tasklist /fi "memusage gt 40000"
```

A final example shows how to clarify the multiple entries for the process "svchost.exe" that occur. (Each has a different PID.)

Service Host (svchost.exe) is a basic piece of the Windows XP OS that is involved with many low-level system services. These are placed in several service groups, all running under the generic service name "svchost.exe" .([See the discussion here.](#)) To

see which services are associated with each instance of svchost.exe, use the command

```
tasklist /svc /fi "imagename eq svchost.exe"
```

More information on Tasklist is at [this Microsoft site.](#)

[Back to top](#)

# The Command Line in Windows

## Site navigation

powered by [FreeFind](#)

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and  
reference](#)

[Commands that everybody  
can use](#)

[Configuring the command  
prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command  
line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the](#)

## Tips for Using the Windows Command Prompt

Here's an assortment of tips and tricks for using the Windows command shell.

There are certain little tricks that books on the command prompt don't always tell you. Or, if they do, the description is buried away in a paragraph somewhere. Experienced users of the command line know all about these. However, average users may not and I am going to mention a few useful tips for them. As far as I know the tips work in both Windows XP and Vista except where noted. They may or may not work for older editions of Windows. Tips for Vista only are given [on another page](#).

### How to make a blank line in a batch file

Sometimes you would like a blank line or two in the output from a batch file. It isn't immediately clear how to do this. Simply entering "echo" doesn't work because that will output the status of command echoing. The trick is to enter

```
echo.
```

Note that "echo" is followed by a period with no space in between.

### Force the "echo" command not to parse arguments

The preceding tip is a special case of a more general method for using the "echo" command. Although the command is used to display text or messages, it can also take certain arguments such as "on" or "off". If you write

```
echo off
```

you will not get a display of the string "off" but will actually be configuring the "echo" command itself. To display the string "off", you would use

```
echo. off
```

In other words, placing a period at the end of "echo" forces the command to simply display whatever follows without checking to see if the string is one of the special cases.

### Check if a file exists

A special variant of the ["if" statement](#) can be used to find out if a file is already present. The statement is

```
if exist somefile somecommand
```

The statement can also test for non-existence of a file with

```
if not exist somefile somecommand
```

### The useful device "nul"

The invisible null device called "nul" has a number of uses. (It's also sometimes called the "bit bucket" or the "black hole".) Anything sent to it disappears. It can be used in statements when you do not want output to be displayed. For example the command

```
somecommand > nul
```

will carry out some command but send whatever is the normal output into oblivion. Sometimes in a batch file you do not want any possible error messages to be displayed. This is done by using

```
somecommand 2> nul
```

[command shell](#)
[Tskill and Taskkill](#)
[Variables and "Set"](#)
[Vista command list](#)
[Vista command line tips](#)
[Xcopy](#)
[Related links](#)
[Computer Education](#)
[Surf the Internet Safely](#)
[Blog with tips](#)
[Windows for Beginners](#)
[Windows Registry](#)

## Stopping a runaway command

Sometimes you start a command only to find that it is going on and on, spewing out screen after screen of output. Most of the time you can terminate a command by simultaneously pressing the two keys "Ctrl" and "c".

## Pausing a scrolling screen

If you have a command with a lot of output,, you can pause the scrolling so that you can read what's on the screen. Use the keyboard combination "Ctrl+s". To resume scrolling, repeat Ctrl+s

## Use drag and drop

Having to type the fully qualified path of a file every time it's needed in a command can be tedious and subject to error. Many people are unaware that a file can be dragged from a folder or Windows Explorer view and dropped on an open command window. It saves a lot of typing. (Doesn't work in Vista)

## Go up one level above the working directory

Any Unix user knows this one but it's often new to Windows users. To go up to the directory that is one level above the working directory, enter

```
cd ..
```

You can repeat this to go up more levels. It's a little off the subject of the command shell but in the Start-Run line just entering the two periods ".." will also take you up one level from the default working directory (the working directory is normally %USERPROFILE%)

## How to change the working directory to a folder on a different drive

If you want to change the working directory for a command window to a folder on a different drive, the command "cd" doesn't work. You have to first enter the drive letter and colon and then enter "cd" and the folder path. However, you can use the switch /d to change the current working directory drive as shown below:

```
cd /d E:\test
```

You can also make the change with one command entry if you use "pushd" instead of "cd":

```
pushd E:\test
```

## Watch out for spaces in file and folder names

The command shell does not recognize spaces in path names. Any path name with spaces must be enclosed in quotation marks. This problem often crops up in scripts where certain environment variables or input arguments are used. For safety, variables that involve file or folder names should be enclosed in quotes.

## Special treatment of variables in "For" statements in batch files

["For" statements](#) are very useful, providing powerful iterative methods. They have the peculiarity, however, of requiring double percent signs for iteration variables in batch files. In other words the syntax in a batch file is:

```
for %%variable In set Do statement
```

If a "For" loop is run directly from the command line, only a single percent sign is used. The syntax is then:

```
for %variable In set Do statement
```

## Case-sensitive variables in "For" statements

In contrast to Unix systems, Windows is usually not case-sensitive. However, iteration variables in "For" statements are case-dependent. So %A is a different variable from %a.

## Pin a command-line shortcut to the Start menu

If you use the command prompt frequently, make it easily accessible. Open Start-All Programs-Accessories and right-click the entry "Command Prompt". Select "Pin to Start menu" from the context menu. Or go to `WINDOWS\system32` and right-click the command shell file `cmd.exe` and select "Pin to Start menu" from the context menu.

## Create a shortcut to a command

If there is a command that you use frequently, you can create a shortcut. The trick is to use the switch `/k` so that the command prompt stays open. The entry for the shortcut should be

```
cmd /k somecommand.exe
```

If the command also needs switches, those can be added as well. (The general details of making a shortcut are [at this page](#).)

## Open Windows Explorer from the command line

To open the current command-line directory in a Windows Explorer window use the command

```
start .
```

To open the directory above the current command-line directory in a Windows Explorer window use the command

```
start ..
```

(Windows XP only) To open *My Computer* in a Windows Explorer window use the command

```
start ...
```

## Using the command "Start"

The tip given above is an example of how the "Start" command can be used to invoke an action or a system folder or an URL. For example, simply entering "cookies" in the Run line will open the system folder Internet Cookies in Windows XP (but not in Vista). However, in the command shell, you would need to enter

```
start cookies
```

In Vista, the command has to be modified with the [Shell command](#) and would be

```
start shell:cookies
```

Similarly, you can open a program like Microsoft Word with the command

```
start winword
```

You can also open a Web page in Internet Explorer with a command of the type

```
start http://somesite.com
```

## Save typing with file-name and folder-name completion (Tab completion)

A very useful feature that can save a lot of typing is the name or path completion function. This feature uses the `Tab` key to complete file and folder names that you begin typing. For example, type "a" (no quotes) into a command line and then press the `Tab` key. Windows will complete your typing with the name of an existing file or folder beginning with "a", starting in alphabetic order. Press `Tab` again and the next possible file or folder will be displayed. In this way, you can cycle through all files and folders existing in your current path that begin with a particular character or group of characters. The keyboard pair `Shift + Tab`

will take you backwards in the list. The tab completion function can be used in more than one place in a command.

### [Enable QuickEdit mode for the command window](#)

Being able to cut and paste to and from the command window is very handy but it is not enabled by default. I use this feature frequently and I suggest that you enable it for all command windows. The details of how to enable QuickEdit are given [on another page](#). Once QuickEdit is enabled, the contents of the clipboard can be entered into a command prompt by right-clicking in the command window.

### [Display the Command History](#)

The default setting for the [configuration of a command window](#) includes the capability for storing up to 50 previously entered commands. The command history can be displayed by entering the "F7" key.

### [Use the "sleep" command in Windows XP batch files](#)

Sometimes it is desirable to have a batch file wait a certain amount of time before it carries out the next command. If you download the free Windows 2003 Server tools ([described on another page](#)), one of the available tools is *sleep.exe*, which provides a way to make batch files wait a specified interval. For an interval of *n* seconds the command is:

```
sleep n
```

### [Copy text from the console window](#)

Way back in the days of DOS, it was not uncommon to enter text directly from the command window into a file with the "copy" command. That is less common in Windows but the capability is still there. Output from the command window or console is denoted by CON. (It is not case-sensitive.) To copy text from the command window to a file "sometext.txt", the sequence of statements would be

```
copy con sometext.txt
First line of your desired text
some more text...
^Z
```

The last line indicates the keyboard combination of the Control key and "z" followed by pressing the Enter key. This command terminates the sequence and sends the text to the desired file, which it creates. This particular example places the file in the working directory but other paths can be used.

## Tips for the Vista command shell

Windows XP and Vista share many of the same features in the command line. However, as to be expected, there are some differences. Tips that are relevant to Vista only are given [on the next page](#).

[Back to top](#)



# The Command Line in Windows

## Managing Windows XP Programs: Tskill and Taskkill

Windows XP comes with several tools for ending programs or processes from the command line. The features and application of Taskkill and Tskill are discussed.

Sometimes it is desirable to end a program or a process from the command line. The process may be hung or not responding or it may be desirable to have a script for ending it. Both the Home and Professional version of Windows XP come with the tool [Tskill](#). In addition, XP Professional has the more powerful tool [Taskkill](#). Although the graphical utility [Task Manager](#) can be used to terminate programs that are hung up, the command line can be faster and easier to use. Also, there may be situations where it is convenient to have a batch file that can be run as a script. In addition, Taskkill is capable of sophisticated filters

### Tskill

The syntax for the command is

```
TSKILL processid | processname [/SERVER:servername] [/ID:sessionid | /A] [/V]
```

The meaning of the various parameters is given in Table I.

Table I. Parameters for the command Tskill

Parameter	Description
processid	PID for process to be terminated. Use only if <i>processname</i> is not used
processname	Process name to be terminated. Wildcards can be used here . Do not use if PID is used
/SERVER:servername	Server containing processID (default is current). Usually not needed on home PCs
/ID:sessionid	End process running under the specified session. Often not needed on home PCs
/A	End process running under ALL sessions (administrator privileges required)
/V	Display information about actions being performed

An example of a simple command that would end Notepad would be

```
tskill notepad
```

Another example is ending all the Microsoft documents that you have open

```
tskill winword
```

All open Word documents will be closed but the contents will not be saved so make sure to save important work. An administrator can close processes that might be running in sessions started by other users. The command

```
tskill winword /a
```

will close everybody's open Word documents.

It may not always be obvious what process name to use for a program. Usually the name of the program executable file (minus the EXE extension) will work. One way is to use [Tasklist](#) to find the PID and use that. Another is to use [Task Manager](#) to find the process associated with a program. (Of course, Task Manager itself can be used to terminate a program.)

### Taskkill

#### Site navigation

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the command shell](#)

[Tskill and Taskkill](#)

[Variables and "Set"](#)

[Vista command list](#)

[Vista command line tips](#)

[Xcopy](#)

## Related links

[Computer Education](#)

[Surf the Internet Safely](#)

[Blog with tips](#)

[Windows for Beginners](#)

[Windows Registry](#)

A tool with more options is provided by [Taskkill](#). The command syntax is

```
TASKKILL [/S system [/U username [/P[password]]]]{ [/FI filter]
[/PID processid | /IM imagename] } [/F] [/T]
```

The various parameters are described in Table II.

*Table II. Parameters for Taskkill command*

Parameter	Description
/S system	Specifies the remote system to connect to. Not needed for most home PCs
/U username	User context under which the command should execute. Often not needed on home PCs
/P password	Password for username
/FI filter	Displays a set of tasks that match criteria specified by the filter
/PID process id	Specifies the PID of the process that has to be terminated. Not used when image name is given in the command
/IM imagename	Specifies the image name of the process that has to be terminated. Wildcard '*' can be used to specify all image names. Not used if PID is given in the command
/F	Forces the termination of all processes
/T	Tree kill: terminates the specified process and any child processes which were started by it

Parameters like the image name or the PID may not be immediately obvious and [Tasklist](#) can be used to obtain them. Taskkill has more options than Tskill and is accordingly more complicated to use. For example, the simple command "Taskkill notepad" won't work. First of all the image name is "notepad.exe" and not the program name "notepad". Also, generally you will have to use the forcing switch. The command to close notepad would be

```
taskkill /im notepad.exe /f
```

Another example is to close down several programs at once.

```
taskkill /f /im notepad.exe /im mspaint.exe
```

The Microsoft literature is not consistent about whether the /f switch goes before or after the image name but it doesn't seem to matter.

## Filtering Taskkill output

Taskkill becomes especially powerful when filters are used with the switch "/fi". Various rules can be formed by using the comparison operators shown in Table III.

*Table III. Comparison operators for filters*

Operator	Description
eq	Equals
ne	Does not equal
gt	Greater than. Only used with numeric values
lt	Less than. Only used with numeric values
ge	Greater than or equal to. Only used with numeric values
le	Less than or equal to. Only used with numeric values

Table IV shows the variables that can be used in a filter.

Table IV. Filter operators and allowed values

Parameter	Valid operators	Valid values
ImageName	eq, ne	Any valid string
PID	eq, ne, gt, lt, ge, le	Any valid positive integer
MemUsage	eq, ne, gt, lt, ge, le	Any valid positive integer in kilobytes
CPUTime	eq, ne, gt, lt, ge, le	CPU time in the format of <i>hh:mm:ss</i> .
Session	eq, ne, gt, lt, ge, le	Session number
Status	eq, ne	Running, Not Responding
Username	eq, ne	Any valid user name (includes SYSTEM, LOCAL SERVICE , NETWORK SERVICE)
WindowTitle	eq, ne	Any valid string
Services	eq, ne	Service name
Modules	eq, ne	DLL name

## Examples of using filters in Taskkill

With filters, you can impose some specific set of conditions that must be met. Filters give Taskkill considerable versatility and allow you to fine-tune the target..Some examples are given below. Note that a specific image name or PID does not have to be included when using filters.

Forcefully shut down all the processes that are not responding. Can be used to make a little batch file to shut down hung or frozen programs.

```
taskkill /f /fi "status eq not responding"
```

Forcefully shut down all programs using a specific DLL file named "some.dll". This should be used with care but one application might be to stop processes thought to be associated with a DLL from spyware or a Trojan. Use [Tasklist](#) to see what processes are using a given DLL.

```
taskkill /f /fi "modules eq some.dll"
```

Close down all programs using large amounts of memory, say 40 MB. Use with care.

```
taskkill /f /fi "memusage gt 40000"
```

Close down programs using more than 40 MB of memory but not Windows Explorer

```
taskkill /f /fi "imagename ne explorer.exe" /fi "memusage gt 40000"
```

[Back to top](#)

# The Command Line in Windows

## Site navigation

[Home](#)
[Assoc and Ftype](#)
[Batch files- basics](#)
[Batch files - branching](#)
[Batch files- iterating](#)
[Command Line-](#)
[Introduction](#)
[Command line list and reference](#)
[Commands that everybody can use](#)
[Configuring the command prompt window](#)
[Doskey](#)
[File system utility- Fsutil](#)
[Net Services](#)
[Netstat](#)
[Network Services Shell](#)
[PowerShell](#)
[Recovery Console](#)
[Recovery Console-](#)
[Commands](#)
[Registry editor console](#)
[Scripts in the command line](#)
[Server 2003 tools for XP](#)
[Service Controller](#)
[Command \(SC\)](#)
[Shell command](#)
[Start-Run line](#)
[Support tools](#)
[Tasklist](#)
[TCP/IP networking tools](#)
[Tips for using the](#)
[command shell](#)
[Tskill and Taskkill](#)

## Variables in the Windows command shell

Declaring variables with the "set" command and their use is discussed.

Variables have a core place in many scripting languages but play a lesser role in the Windows command line. Many commands are predefined and the scope of variables is rather limited. Nonetheless, there are important applications of the command line where variables must be employed and in this article I will outline how the command line uses variables.

### How variables are defined with the "set" command

In one sense, there are two categories of variables for the command line. Some might use the term "variable" for the placeholders or arguments %1, %2, ..%9, that are used to represent user input in batch files. ([See the discussion on this page.](#)) However, the term "variable" is normally reserved in command line usage for entities that are declared as [environment variables](#) with the "set" command. Note that this is a pretty primitive way to define variables. For example, there is no typing. Environment variables, including numbers, are stored as strings and operations with them have to take that into account. Variables are declared and given a value in a single statement using "set". ..The syntax is:

```
set some_variable = some_value
```

Variable names are not case-sensitive and can consist of the usual alphanumeric and other common characters. Some characters are reserved and have to be escaped. They should be avoided. These include the symbols [in Table II on this page.](#) Also, since these are environment variables, their names should be enclosed in percent signs when used in references and expressions, e.g,

```
%some_variable%
```

. The percent signs are not used in the left side of the set statement that declares a variable.

### Localizing variables

The declaration of a variable lasts as long as the present command window is open. If you are using a batch file that does not close its instance of the command window when the batch file terminates, any variables that the batch file declares remain. If you wish to localize a variable to a particular set of statements, use the "setlocal" and "endlocal" commands. Thus, to confine a variable declaration to a particular block of code, use:

```
....
setlocal
set some_variable = some_value
...some statements
endlocal
...
```

### Variables from user input

The "set" command can also accept input from a user as the value for a variable. The switch "/p" is used for this purpose. A batch file will wait for the user to enter a value after the statement

```
set /p new variable=
```

[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

When the user has entered the value, the script will continue. A message string to prompt for input can also be used. For example:

```
set /p new_variable="Enter value "
```

Note the space at the end of the prompt message. Otherwise, the prompt message and the user-entered value will run together on the screen. It works but it looks funny. The user may be tempted to hit the spacebar, which adds a leading space to the input value.

## Arithmetic operations

The command line is not designed for handling mathematical functions but it is possible to do some very simple integer arithmetic with variables. A switch `/a` was added to the `set` command to allow for some basic functions. Primarily, the use is adding and subtracting. For example, it is possible to increment or decrement counters in a loop. In principle, it is also possible to do multiplication and division, but only whole numbers can be handled so the practical use is limited. Although variables are stored as strings, the command interpreter recognizes strings that contain only integers, allowing them to be used in arithmetic expressions. The syntax is

```
set /a some_variable={arithmetic expression}
```

The four arithmetic operators are shown in Table I. (I have omitted a "modulus" operation, which uses the `%` sign in yet another way. In my opinion this just adds difficulty to an already quirky syntax. Using `%` in more than one sense can only confuse.)

Table I. Arithmetic operators

Symbol	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division

Here is an example of a variable `%counter%` being incremented:

```
set /a counter=%counter%+1
```

This can also be written as:

```
set /a counter+=1
```

## Variables in comparison statements in batch files

Variables are often used in comparisons in conditional statements in batch files. Some of the comparison operators that are used are given in Table I of the [page on "If" statements](#). Because of the somewhat loose way that the command line treats variables, it is necessary to be careful when comparing variables. For strings, the safest way is to quote variables. For example:

```
if "%variable1%" == "%variable2%" some_command
```

[Back to top](#)

# The Command Line in Windows

## Windows Vista Command Line List and Reference

### Site navigation

[Home](#)
[Assoc and Ftype](#)
[Batch files- basics](#)
[Batch files - branching](#)
[Batch files- iterating](#)
[Command Line-](#)
[Introduction](#)
[Command line list and reference](#)
[Commands that everybody can use](#)
[Configuring the command prompt window](#)
[Doskey](#)
[File system utility- Fsutil](#)
[Net Services](#)
[Netstat](#)
[Network Services Shell](#)
[PowerShell](#)
[Recovery Console](#)
[Recovery Console- Commands](#)
[Registry editor console](#)
[Scripts in the command line](#)
[Server 2003 tools for XP](#)
[Service Controller](#)
[Command \(SC\)](#)
[Shell command](#)
[Start-Run line](#)
[Support tools](#)
[Tasklist](#)
[TCP/IP networking tools](#)
[Tips for using the](#)
[command shell](#)
[Tskill and Taskkill](#)

The list of commands available in the command line shell for Windows Vista is similar to that for Windows XP but with some additions. The commands and a brief explanation of their functions are given. Some tips for their use are given [on another page](#).

### Commands in Windows Vista

Originally, Microsoft intended to incorporate a new command shell in Vista but then decided to make the new shell a stand-alone application. The new application is called Powershell and is described [on another page](#). Thus, the old command interpreter *cmd.exe* has been continued although in a slightly newer version. (The Vista version is 6.0 whereas the XP version is 5.1.) Several commands are now in the standard list that were formerly available only in [Support Tools](#) or the [Server 2003 Tools](#). For example, *Robocopy* from Server 2003 is now included. There are also some new commands and they are indicated with an asterisk in the list below. Note that some commands may require administrator privileges. Running a command prompt as administrator is described [on this page](#). Some commands involving specialized system or network administration have been omitted.

ASSOC Displays or modifies file extension associations.

ATTRIB Displays or changes file attributes.

BREAK Sets or clears extended CTRL+C checking.

\*BCDEDIT Sets properties in boot database to control boot loading.

CACLS Displays or modifies access control lists (ACLs) of files.

CALL Calls one batch program from another.

CD Displays the name of or changes the current directory.

CHCP Displays or sets the active code page number.

CHDIR Displays the name of or changes the current directory.

CHKDSK Checks a disk and displays a status report.

CHKNTFS Displays or modifies the checking of disk at boot time.

\*CHOICE Batch file command that allows users to select from a set of options.

CIPHER Displays or alters the encryption of directories [files] on NTFS partitions.

\*CLIP Redirects output of another command to the Windows clipboard.

CLS Clears the screen.

CMD Starts a new instance of the Windows command interpreter.

\*CMDKEY Creates, lists and deletes stored user names and passwords or credentials.

COLOR Sets the default console foreground and background colors.

COMP Compares the contents of two files or sets of files.

COMPACT Displays or alters the compression of files on NTFS partitions.

CONVERT Converts FAT volumes to NTFS. You cannot convert the current drive.

COPY Copies one or more files to another location.

DATE Displays or sets the date.

DEFRAG Disk defragmenter accessory.

DEL Deletes one or more files.



[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

DIR Displays a list of files and subdirectories in a directory.

DISKCOMP Compares the contents of two floppy disks.

DISKCOPY Copies the contents of one floppy disk to another.

DISKPART Displays or configures Disk Partition properties.

DOSKEY Edits command lines, recalls Windows commands, and creates macros.

DRIVERQUERY Displays current device driver status and properties.

ECHO Displays messages, or turns command echoing on or off.

ENDLOCAL Ends localization of environment changes in a batch file.

ERASE Deletes one or more files.

EXIT Quits and closes the command shell.

EXPAND Expands one or more compressed files.

FC Compares two files or sets of files, and displays the differences between them.

FIND Searches for a text string in a file or files.

FINDSTR Searches for strings in files.

FOR Runs a specified command for each item in a set.

\*FORFILES Selects files in a folder for batch processing.

FORMAT Formats a disk for use with Windows.

FSUTIL Displays or configures the file system properties.

FTYPE Displays or modifies file types used in file extension associations.

GOTO Directs the Windows command interpreter to a labeled line in a batch program.

GPRESULT Displays Group Policy information for machine or user.

GRAFTABL Enables Windows to display an extended character set in graphics mode.

HELP Provides Help information for Windows commands.

\*ICACLS Display, modify, backup, or restore ACLs for files and directories ([more here](#)).

IF Performs conditional processing in batch programs.

LABEL Creates, changes, or deletes the volume label of a disk.

MD Creates a directory.

MKDIR Creates a directory.

\*MKLINK Creates Symbolic Links and Hard Links

MODE Configures a system device.

MORE Displays output one screen at a time.

MOVE Moves one or more files from one directory to another directory.

OPENFILES Queries, displays, or disconnects open files or files opened by network users.

PATH Displays or sets a search path for executable files.

PAUSE Suspends processing of a batch file and displays a message.

POPD Restores the previous value of the current directory saved by PUSH.D.

PRINT Prints a text file.

PROMPT Changes the Windows command prompt.

PUSH.D Saves the current directory then changes it.

RD Removes a directory.

RECOVER Recovers readable information from a bad or defective disk.

REM Designates comments (remarks) in batch files

REN Renames a file or files.

RENAME Renames a file or files.

REPLACE Replaces files.

RMDIR Removes a directory.



ROBOCOPY Advanced utility to copy files and directory trees

SET Displays, sets, or removes environment variables for current session.

SETLOCAL Begins localization of environment changes in a batch file.

SETX Sets environment variables.

SC Displays or configures services (background processes).

SCHTASKS Schedules commands and programs to run on a computer.

SHIFT Shifts the position of replaceable parameters in batch files.

SHUTDOWN Allows proper local or remote shutdown of machine.

SORT Sorts input.

START Starts a separate window to run a specified program or command.

SUBST Associates a path with a drive letter.

SYSTEMINFO Displays machine specific properties and configuration.

\*TAKEOWN Allows an administrator to take ownership of a file ([more here](#)).

TASKLIST Displays all currently running tasks including services.

TASKKILL Kill or stop a running process or application.

TIME Displays or sets the system time.

\*TIMEOUT Pauses the command processor for the specified number of seconds. [More here](#).

TITLE Sets the window title for a CMD.EXE session.

TREE Graphically displays the directory structure of a drive or path.

TYPE Displays the contents of a text file.

VER Displays the Windows version.

VERIFY Tells Windows whether to verify that your files are written correctly to a disk.

VOL Displays a disk volume label and serial number.

\*VSSADMIN Volume Shadow Copy Service administration tool

\*WHERE Displays the location of files that match a search pattern.

XCOPY Copies files and directory trees.

WMIC Displays WMI information inside interactive command shell.

[Back to top](#)

# The Command Line in Windows

## Tips for Using the Vista Command Shell

Command-line tips specific to Windows Vista are given.

For the most part, the features of the command line are the same in Vista as they are in Windows XP but there are some differences. (The Vista version is 6.0 whereas the XP version is 5.1.) Here are some tips that apply only to Vista.

### Run as administrator in Vista

The Windows Vista operating system has a security feature called [User Account Control](#) that limits the privileges of users by default. Since the command line is usually involved with administrative tasks, you'll often want to run as an administrator. This can be done each time by right-clicking the icon for the command prompt and choosing "Run as" from the context menu. A more convenient way is to set the command prompt to run with administrator privileges by default. Right-click the command prompt shortcut icon and choose "Properties" from the context menu. Click the "Advanced" button and put a check by "Run as administrator". Click "OK". Note that you will still get the UAC message when you open the command prompt.

Another way to run the command prompt as administrator is to enter "cmd" in *Start Search* and then use the keyboard combination *Ctrl+Shift+ Enter*.

### "Open Command Window Here" in Vista

It is now easy to open a command prompt referenced to a folder of your choice in Vista. If the *Shift* key is held down while right-clicking a folder, the context menu will contain an entry, "Open Command Window Here". Selecting this entry will open a command prompt with the chosen folder as the reference point for commands.

### Open command window with administrator privileges anywhere

The "Run as administrator" option mentioned above always opens with `\\Windows\System 32\` as the working directory. To open a command console with administrator privileges in any directory of choice, you can add a command to the right-click context menu. The INF file to make the appropriate Registry edit can be [downloaded here](#). It is from the PowerToy utility [described at this link](#).

### Send command output to the Windows clipboard with clip.exe

Vista comes with a command-line utility `clip.exe` that can be used to redirect or pipe the output of another command to the Windows clipboard. The command uses a "pipe" and has the form:

```
somecommand | clip
```

For example, to send a directory listing to the clipboard, the command is:

```
dir | clip
```

## Site navigation

powered by [FreeFind](#)

[Home](#)

[Assoc and Ftype](#)

[Batch files- basics](#)

[Batch files - branching](#)

[Batch files- iterating](#)

[Command Line-](#)

[Introduction](#)

[Command line list and reference](#)

[Commands that everybody can use](#)

[Configuring the command prompt window](#)

[Doskey](#)

[File system utility- Fsutil](#)

[Net Services](#)

[Netstat](#)

[Network Services Shell](#)

[PowerShell](#)

[Recovery Console](#)

[Recovery Console-](#)

[Commands](#)

[Registry editor console](#)

[Scripts in the command line](#)

[Server 2003 tools for XP](#)

[Service Controller](#)

[Command \(SC\)](#)

[Shell command](#)

[Start-Run line](#)

[Support tools](#)

[Tasklist](#)

[TCP/IP networking tools](#)

[Tips for using the command shell](#)[Tskill and Taskkill](#)[Variables and "Set"](#)[Vista command list](#)[Vista command line tips](#)[Xcopy](#)

## Related links

[Computer Education](#)[Surf the Internet Safely](#)[Blog with tips](#)[Windows for Beginners](#)[Windows Registry](#)

## [Place the contents of a text file into the Windows clipboard with clip.exe](#)

The utility *clip.exe* can also be used to read a text file and place its contents in the Windows clipboard. The command has the form:

```
clip < somefile.txt
```

## [The batch file command called "choice" is back in Vista](#)

Oldtimers will remember that DOS had a command for batch files called "choice" that allowed for some limited user interaction. The command was then removed from 32-bit command shells because the "set /p" option gave equivalent or better functionality. However, "choice" is back in Vista in a new form. Enter "choice /?" in a Vista command prompt for details about its features.

## [Use the "choice" command to make a Vista batch file wait](#)

One useful application of the "choice" command is to make a batch file pause for a specified period of time. The statement has the form:

```
choice /T n /D y > nul
```

The switch "/T n" specifies a wait period of *n* seconds. The switch "/D y" creates a default choice of "yes". To suppress the unwanted text output of the command, it is redirected to the null device (nul).

## [Use the "timeout" command to make a Vista batch file wait](#)

Another new command in Vista is "timeout". It will cause the command processor to wait for a specified number of seconds or until a key is pressed. The format is

```
timeout /T n
```

where *n* is the number of seconds to wait. To make the command ignore any key presses, the switch */nobreak* can be added:

```
timeout /T n /nobreak
```

Because the command gives output listing the time remaining, it may be necessary to use a redirect to nul.

```
timeout /T n > nul
```

## [Switch added to "Dir" to enable viewing Alternate Data Streams](#)

NTFS files [can have added information](#) in "streams" or "forks". These added items are normally hidden from access by most Windows functions such as Explorer. In Vista a switch */R* has been added to the "dir" command that allows alternate data streams to be listed.

## [Enable the built-in master administrator account on the log-in screen](#)

Vista contains a master administrator account but it is not normally visible on the log-in screen. To enable it, open a command window with administrator privileges and use the command

```
net user administrator /active:yes
```

(Make sure that you assign a password to the account.) To remove the account from the log-in screen, use the command

```
net user administrator /active:no
```

## [Reduce the space used by System Restore](#)

System Restore can use up to 15% of a hard drive for its backup files (shadow storage). As hard drives get ever bigger, that becomes a lot of space. The command "vssadmin" can be used to administer settings for System Restore. To control the space allocation, open a command prompt with administrator privileges and enter

```
vssadmin Resize ShadowStorage /For=C: /On=C: /Maxsize=[n]
```

For [n] enter the desired size in MB or GB. The units must be stated: for example, "Maxsize=500MB" or "Maxsize=2GB". The example is for the C: drive. WARNING! This will delete all your old Restore Points!

## Increase the file system memory cache

If you tend to open and close a lot of files, you may be able to increase performance by creating a larger value for a special cache setting with the [file system utility](#) command

```
fsutil behavior set memoryusage 2
```

[According to Microsoft](#), this increases something called the "paged pool" memory. Do not use if you are already consuming large amounts of system memory with other activities. If performance after the change is unsatisfactory, undo it with the command

```
fsutil behavior set memoryusage 1
```

These commands require administrator privileges. The commands change a Registry setting and take effect after a reboot.

## Use "takeown" to access certain files

Vista protects many system files for security reasons and even an administrator is not allowed to access them. If you are denied access to a file while in an administrator account, you can use the command line tool "takeown.exe" to reassign ownership. You will need to run the command from a command prompt with administrator privileges. The syntax is

```
takeown /f some_file [/a] [/r]
```

The specified filename can contain wildcards. You can also specify a folder. The optional switch "/a" transfers ownership to the administrators group. If omitted, the default is to transfer ownership to the present user account. The switch "/r" recurses subdirectories. Although this command assigns ownership, it does not give control rights. Thus if you wish to modify a system file (often not a good idea) you will probably have also to apply the "icacls" command discussed next.

## Obtain control rights to a file with "icacls"

(Icacls.exe supersedes the "cacls" command of Windows XP. The older command is still available, however.) This command has a rather complex set of options. They can be displayed by entering "icacls /?" in a command prompt. One example is the command to grant full access rights to an account named *user*:

```
icacls file_name /grant user:F
```

## Clean up Vista SP1 files

When you install Vista service pack 1, a facility for uninstalling it is also created. If you have SP1 installed for a while and are satisfied that you will keep it, you can remove the uninstall files and free up almost a GB of disk space. To remove the backup files, use the command

```
VSP1CLN.EXE
```

(I have used caps to make the difference between "one" and "ell" clear but case doesn't matter.) Administrator privileges are required. After running this command, you will be unable to uninstall Vista SP1 so be sure you really want to keep it.

## Using drag and drop- not

The useful capability to drag commands and drop them into a command prompt that was present in past versions of Windows does not work in Vista. (However, it has been restored in Windows 7.)

[Back to top](#)