Introduction to Windows Command Line Console

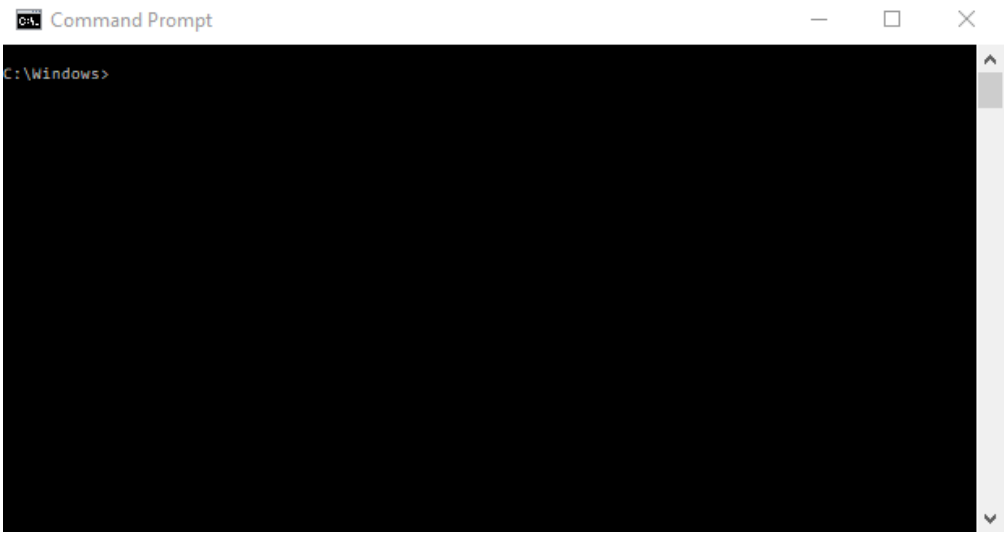
This document assumes you are using Windows 11.

Before Windows was created, the most common operating system that ran on IBM PC compatibles was DOS (Disk Operating System). DOS was a purely textual operating system, i.e. in order to run programs or manipulate the operating system you had to type in commands. When Windows was first created it was actually a graphical user interface that was created in order to make using the DOS operating system easier for a novice user. As time went on and newer versions of Windows were developed DOS was finally phased out with Windows ME. The newer operating systems do not run on DOS. However, they do have a "command prompt", which has a similar appearance to DOS. The command prompt in Windows XP / 7 /8 / 8.1 / 10 / 11 is NOT a "DOS window". Windows 10 is a 64-bit protected memory system with a totally different approach from the DOS/9X/Me family. The command line in Windows has many more capabilities and none of the 16-bit limitations. The augmented capabilities make the command line a powerful tool.

The command prompt window

The command prompt (also known as the Windows Console) is run from its own window by invoking the Windows 11 command interpreter. To open the Windows Console, type "cmd" into MagnifyingGlass-Run box in the taskbar or open the command prompt through Start- All Apps - Windows Tools - Command Prompt.

You will be presented with a window that looks similar to below.



The command prompt is simply a window that by default displays the current directory (or “folder”). It has a blinking cursor ready to accept typed in commands. For example, above you can see that it says C:\WINDOWS>. The C:\WINDOWS> is the prompt and says that you are currently in the c:\windows directory.

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

WINDOWS INTERNAL COMMANDS

assoc	dir	move	set
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

A list of the commands available can be obtained by entering “help” without quotes into the Windows Console.

To get more detail on a particular command, type

help command-name

or

command-name/?

into the Windows Console.

For example

C:\> help cls

Displays the information

"Clears the screen"

The command

C:\> cls /?

Displays the same information.

Some Basic commands

The Command cd

This command allows you to change your current directory or see what directory you are currently in.

C:\> cd \
C:

C:\> cd Windows
C:\Windows>

C:\Windows> cd System32
C:\Windows\System32>

Use the command line console to find more information on the command **cd**.
(Hint: Help CD)

When using the **cd** command you must remember how paths work in Windows. A path to a file is always the root directory, which is symbolized by the standalone **** symbol, followed by the directories underneath it. For example, the file notepad.exe which is located in c:\windows\system32 would have a path as follows c:\windows\system32\notepad.exe.

Question: What does **cd..** mean?

Assuming you have a second drive available labelled **D** (for example, if you plugged in a USB drive), the command to change to that drive from the command line is:

C:\Windows\System32> D:

Now, assume you are in the folder **C:\Windows\System32** and you want to change to the folder **C:\Windows\Logs** How do you do that?

C:\Windows\System32> cd ..\Logs
Note: there is a space betwee the CD and the double dots.

The Exit **command**
This command will close the command prompt. Simply type exit and press enter and the command prompt window will close.

The dir **command**
This command will list the files and directories contained in your current directory, if used without an argument, or the directory you specify as an argument.

Type **dir** into the command line now. As well as the list of files what other information do you see?

You should notice that there are two directories named . and ..
These directories have a special meaning in operating systems.
The . stands for the *current* directory and,
the .. stands for the *previous* directory in the path.

Using the Wildcard Symbols * and ?

In the Windows Command Prompt (cmd), the wildcard ***** is used as a placeholder to represent multiple characters in filenames or directory names. It's a powerful tool for searching, listing, or manipulating groups of files that share a common pattern. Here's how it works:

- *** (Asterisk)**: It matches any number of characters, including zero characters.
- **? (Question mark):** it matches exactly one character.

Examples of using the Asterisk wildcard:

1. Listing files with a specific extension:
   Command: dir *.txt
   Explanation: Lists all files in the current directory that end with the .txt extension. The * here matches any file name, as long as it ends with .txt

2. Listing files that start with a specific letter:
   Command: dir a*
   Explanation: This lists all files in the current directory that start with the letter "a" followed by zero or more characters.

3. Listing all files:
   Command: dir *
   Explanation: This lists all files and directories in the current directory, because * matches everything.

4. Renaming multiple files:
   Command: rename *.txt *.bak
   Explanation: This command renames all files with the .txt extension to the same filenames with the .bak extension.

The * wildcard can be used in various commands like dir, copy, del, rename, etc. It works not just for file extensions, but also for parts of filenames, so it's very flexible in matching patterns.

Examples of using the ? wildcard:

1. Match Files with a Single Character Difference:
   Command: dir file?.txt
   Explanation: This matches files like file1.txt, fileA.txt, and fileZ.txt, but not file.txt (because it's missing a character) or file12.txt (because it has an extra character).

2. Match Files with Multiple Variations:
   Command: dir file???
   Explanation: This will match files like file123, fileabc, or fileXYZ, where there are exactly three characters after file.

3. Match Files with Specific Start and End Characters
   Command: dir a?c*.docx
   Explanation: This will match filenames that start with the letter "a", have any single character in the second position (because of ?), followed by "c", followed by any sequence of zero or more characters (because of *) and having the extension .docx.
   For example, this would match abc123.docx, acc.docx, and a1c_report.docx.

The Copy **command** - This command allows you to copy files from one location to another. For example:

copy sourcefile destinationfile.

For example if you have the file c:\test\test.txt and would like to copy it to c:\windows\test.txt then type

copy c:\test\test.txt c:\windows\test.txt

If the copy is successful it will tell you so and give you back the prompt. If you are copying within the same directory you do not have to use the path. Here are some examples and what they would do:

copy test.txt test.bak

Copies the **test.txt** file to a new file called **test.bak** in the same directory

copy test.txt c:\temp

Copies the **test.txt** file to the **\temp** folder on the current drive (the c:\temp folder must already exist).

The Move **command** - This command allows you to move a file from one location to another. Examples are below:

move test.txt test.bak

Moves the **test.txt** file to a new file renaming it to **test.bak** in the same directory

move test.txt h:\temp

Moves the test.txt file to the \temp directory on the h: drive

move * \windows

Moves all the files in the current directory to the \windows directory on the current drive.

Symbols

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.

List of 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 if 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, "|" .

A frequent use of the redirection is to save some output to a text file. For example, the command **dir myfolder > myfile.txt** sends a list of the files in "myfolder" to a text file "myfile.txt".

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". For example:

```
dir myfolder | more
```