What is DOS and Why is it Needed?

In general, "every computer requires an operating system." This applies to the smallest, dedicated processor to most advanced supercomputer. The operating system provides exactly what it's name imp lies: control over *operations* and resources, with such operations being,

- execute (run) software (specific tools for the user)
- interact with peripheral devices (keyboards, printers, video)
- store and retrieve data correctly to/from secondary storage devices (disk, tape, off-board memory)
- provide an application-interface (for resources), and user-interface

The first popular home computers (and even the original IBM-PC) had a "built-in Microsoft BASIC" (in ROM) that performed two functions: a basic operating system (with default secondary storage being audio tape), and an interactive/interpreted general-use programming language (BASIC). The interaction to the computer was accomplished through the command line, with operating system and programming instructions indistinguishable from each other.

With the introduction of floppy diskette drives (and later harddisk drives), the Microsoft Disk Operating System (MS-DOS; originally PC-DOS) became a superset of the existing MS-BASIC instructions. It provided access to the disk technologies, access to the increased memory of new machines (originally MS-BASIC could access only 64KB), and provided extra BASIC instructions for disk-access, memory data-structures, and graphics.

For the first versions of MS-DOS/PC-DOS on disk (compared to MS-BASIC in ROM)

- advantage: DOS was not built-in to hardware, but loaded from disk with user-configurable boot settings.
- advantage: upgrades to the OS are simple, done by just changing (or replacing) the software on disk; from applications to the entire operating system

- disadvantage: without DOS, the system cannot access diskette drives, and reverts to built-in MS-BASIC, which only uses tape (all modern computers allocate space for MS-BASIC, but do not actually include it in memory).
- disadvantage: significantly slower boot time, since everything must be loaded from disk, versus the "instant on" boot from MS-BASIC in ROM)

Are there any other advantages and disadvantages? Do not forget about ideas such as licensing and illegal copies.

The idea of an "OS on a chip" is not completely dead, and has become popular once again. Handheld computers utilise operating systems, such as PalmOS, PocketPC (WinCE), and PortableLinux, that are loaded instantly, upgradeable, graphical, and configurable. Some manufacturers are even considering placing WindowsXP (or the next version) or Linux on a chip for notebooks computers and tablet PCs, and special "turn key" server systems that boot in seconds rather than minutes.

Fundamental Files

There are three (3) fundamental files to MS-DOS (more are necessary for Win9x, and even more for Windows NT/2000/XP), which are loaded in the following order:

- IO.SYS - extends the system's existing hardware BIOS
- MSDOS.SYS - loads basic disk and memory access utilities (the purpose of this file changes if the OS is booted to the GUI)
- COMMAND.COM - the command line interface, command interpreter, and application interface (sometimes referred to as the DOS "kernel")

Whenever a command is typed at the DOS prompt, it is read and interpreted by COMMAND.COM, this is the program that interacts with the user at the "command prompt."

Types of Commands

All DOS commands are divided into two categories: *interneral* and *external*

Internal

- are part of COMMAND.COM, and therefore stored in memory
- available all the time
- do not exist as individual (executable) program files on disk
- tend to be small programs (in terms of code size)

examples: DIR, CD, COPY, DEL, VER

External

- reside on disk (somewhere), loaded when required
- there is no difference between these programs and third-party applications (which include programs you write)
- also called "DOS Utilities"
- some external commands are loaded and *kept* in memory, these are called "terminate-stay-resident" (TSR)

examples: FORMAT, FDISK, XCOPY, MOVE; TSR's: DOSKEY, SUBST

Why are both *internal* and *external* commands defined; why not make them all internal or external?

What defines whether one command is internal or external?

Using Commands

```
{command} {switches} {parameters} {more switches}
```

Each command has its own switches and parameters, since each command has a different purpose.

An important switch is **help /?**

example: DIR /? gives help on the DIR command

Filenames

As with any operating system, filenames are used to identify the starting position in a storage device of a specific data file. DOS follows standard filename conventions, in that the name is formed of two parts: a *filename* and *extension*, with a period, or "dot," used to separate the two.

The *filename* is used to uniquely identify the "content" of the data file, whereas the *extension* is used to identify the "data type" of the data file. For example, as content the file **REVQRTR1.xls** holds the "revenue from quarter 1," and it is a "Microsoft Excel" type of file (because of the .xls extension). Many modern operating systems use the extension to know which application the file belongs to; few operating system actually read the file to understand its type.

Unlike other modern operating systems, such as VAX/VMS, UNIX/Linux, Windows, and MacOS, DOS places a few restrictions on the filename,

- the following symbols are not valid for naming: *, ?, . , |, \, /, <, >, comma, space
- *filename* size is between 1 and 8 characters/symbols, and there must be at least one character/symbol
- *extension* size is between 0 and 3 characters; optional, but necessary to determine *file type*
- no distinction is made between UPPER- and lowercase characters

Further, no two files can have the same full name, meaning the same filename and extension. For operating system that seem to, a "version number" is used to distinguish files (such is used with VAX/VMS).

Wildcards

Many DOS commands can perform a single action over a selection of files in a group; this saves time, keystrokes, and user aggravation. Further, this is one of the most powerful features of DOS commands: *a single instruction effecting a multitude of files*.

In order to specify which files belong to the group, special characters to specify the types of files to group. These characters are called "wildcards," because they take the place of any symbol in the filename (just like poker).

- **?** – specifies a single character in the filename or extension
- ***** – specifies any group of characters in the filename or extension

Consider the following examples with the command **xcopy**,

- **move *.* a:** – move files with any filename and extension from the current directory to A:\
- **xcopy file?.d* d:*.e*** – copy files starting with "file" + any single character, extension starting with "d" to D:\ and rename the extension to start with "e" *(note: this command copies and renames!)*
- **del bob*f.** – delete all files starting with "bob" and ending with "f", and with no extension

There are many tricks and uses of wildcards in grouping files together for one command. With the use of wildcards, a single DOS command, like **xcopy**, can accomplish an action that would take a user several mo use clicks with a GUI.

Directories

This topic is left to the **DOS** workshop, and the experience gained from using directories.

Other Topics and Personal Research

With the popularity of graphical operating systems, command-line systems have become viewed as out-dated and obsolete. This is true of only one modern operating system: Apple MACOS, which traditionally has never command line, but in the latest version (MacOS 10.x), it has a powerful UNIX-like command line.

UNIX/Linux, VAX/VMS, Windows 9x/Me/NT/2000, and Novell Netware are all operating systems that have a graphical interface, which rests on an older, reliable command-line OS. There are many powerful aspects of these systems that are hidden because of the graphical interface.

Studying DOS should be an area of personal study to learn and understand one of the oldest and important operating system interfaces.