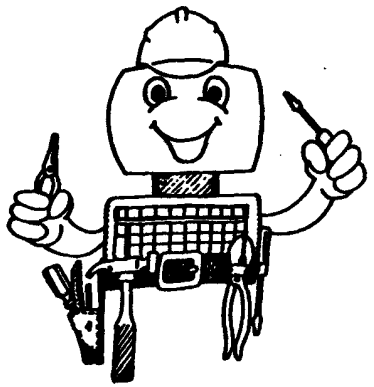
LEVEL II

TOOLS



LICENSE AGREEMENT

Level II Tools and the accompanying documentation are governed by the terms of the following license agreement.

You, the user, are granted a paid up license to use this software package and accompanying documentation for an unspecified amount of time. You in no way become the owner of the package, nor do you have the right to sell or re-sell the package. You are allowed to make copies of this package for archival use only, any other copying is a violation of this agreement.

Help keep the price of software down...DON'T PIRATE!

DISCLAIMER

Although this software package is designed to prevent the accidental loss of data, the deleting of important files is still possible.

The author of this software is in no way responsible for the loss of time, money, or data as a direct or indirect result of the use of this software package.

Level II Tools

Release 1.00

Copyright (c) 1989 By Keith J. Alphonso
Licensed to: Alpha Software Technologies
All Rights Reserved

INTRODUCTION

The OS9 Toolkit is a set of utilities that are useful in the day-to-day usage of a computer. They allow the computer to help you do your work so that you are free to do other things. They also assist you in your work by giving you various features and information. I find myself using these tools all of the time, and I sincerely hope that you find them just as useful.

OVERVIEW

The OS9 Toolkit will be of most use to you if you copy all of the tools it contains to your working OS9 directory. The copying of these commands is explained in the installation section of this manual. Once you have copied all of these commands to your working OS9 disk, you can use the commands at any time. This is necessary as you will probably want to use them often.

INSTALLATION

REQUIREMENTS:

Radio Shack Color Computer
At least one disk drive.
OS9 Operating System

MAKE A BACKUP

The first thing that you should do is make a backup of the distribution disk. To do this follow these steps:

1> Get a new blank diskette
2> Format the new diskette (Format /d0)
3> Backup the OS9 Toolkit disk (Backup /d0 #40k)

COPY THE TOOLKIT TO YOUR WORKING DISKETTE:

Once you have a backup of your Toolkit diskette you are ready to copy the files to your working disk. Follow these steps:

(For users with 1 disk drive)
1> Boot up with your normal boot disk
2> Insert the OS9 Toolkit disk.
3> Type 'chd /d0'
4> Type 'INSTALL1'

(For users with 2 disk drives)
1> Boot up with your normal boot disk
2> Insert the OS9 Toolkit in drive 0
3> Type 'chd /d0'
4> Type 'INSTALL2'

WILD CARD COMMANDS

A wildcard is a character ('*') which can be used to specify multiple characters. The wildcard character is placed in a file name to represent that any character, characters or no characters can take its place. This can be very helpful when more than one file needs to be copied, deleted, attribed, or matched.

The following are some example uses of wildcard characters and what file names they would match.

The wildcard name '*' describes ALL filenames.

The wildcard name 'foo*' describes all filenames that begin with 'foo'. This could include 'football', 'fooey', 'fooface', 'foo', or any other filename that begins with 'foo'.

The wildcard name '*foo' describes all filenames that end with 'foo'. This could include 'tufoo', 'wafoo', 'whofoo', 'foo', etc.

The wildcard name 'foo*foo' describes all filenames that begin and end in 'foo'. This would include 'foowhofoo', 'foomanfoo', 'foofoo', etc.

The wildcard name '*foo*' describes all filenames that have the word 'foo' anywhere in them, such as 'football', 'tufoo', 'whofoomu', 'foo', etc.

Wild card COMMANDS are used to manipulate multiple files at one time. Wild card commands work almost identically like their non wild card counterparts. These commands are Wcopy, Wdel, Wattr, and Wmatch; they are similar to copy, del, attr, and dir respectively. The following is a detailed description of each command.

Wcopy

The WildCopy command (Wcopy) will copy multiple files from one directory to another. The syntax is as follows:

Wcopy <source> [<destination>]

All files matching the <source> specification will be copied to the <destination> directory.

The <source> can contain a wildcard character. It can also contain an entire pathlist (as in '/d0/sources/asm/*.a' or a simple file name (as in '*.a'). If a complete pathlist is not included the current directory is assumed.

The <destination> cannot contain a wildcard character. It should only contain a directory name to which files will be copied. The brackets ([]) around <destination> mean that it is optional. If you do not specify a destination directory, the current directory is assumed.

If the specified file already exists you will be asked if you want to overwrite the file. If you do simply type 'Y' and the file will be overwritten.

If the specified directory is full you will be asked to insert another disk and press a key. Simply insert another diskette and press a key. This allows the copying of directories that exceed the size of one diskette.

The following are some examples of Wcopy:

Wcopy /d0/sources/asm/*.a /d1
Wcopy /d0/sources/asm/*.a
Wcopy *.a /d0/sources/asm

NOTE: You should never use 'Wcopy *' by itself. This will cause Wcopy to assume that the current directory is the source as well as the destination directory. When you answer 'Y' to the overwrite prompt the file will be deleted. An ERROR 216 will then be generated when the copy is again attempted. THIS RESULTS IN THE ORIGINAL FILE BEING DELETED!!

NOTE: If during the copying process a sub-directory is found an 'Access denied!' will be generated. Simply ignore this error message.

NOTE: Because this program calls the copy command (that comes with OS9), COPY must be present either in memory or in the current execution directory.

Wdel

The WildDelete command (Wdel) will delete multiple files from a directory. The syntax is as follows:

Wdel <pathname>

All files matching the specified <pathname> will be deleted.

The <pathname> can contain wildcard characters. It can also specify a complete pathlist (as in '/d0/sources/asm/*.a') or a simple file name (as in '*.a').

NOTE: Use caution when using Wdel as it is very easy to wipe out an entire directory.

Wattr

The WildAttr command (Wattr) will set the attributes to multiple files in a directory. The syntax is as follows:

Wattr <pathname> <attribute_list>

All files matching the specified <pathname> will be given the attributes specified in <attribute_list>.

The <pathname> can contain wildcard characters. It can also specify a complete pathlist (as in '/d0/sources/asm/*.a') or a simple file name (as in '*.a').

The <attribute_list> is a list of attribute specifications for a file. The specification list is passed directly to the ATTR command that comes with OS9. The list of specifications is as follows:

r	- set read attribute
-r	- reset read attribute
w	- set write attribute
-w	- reset write attribute
e	- set execution attribute
-e	- reset execution attribute
pr	- set public read attribute
-pr	- reset public read attribute
pw	- set public write attribute
-pw	- reset public write attribute
pe	- set public execute attribute
-pe	- reset public execute attribute
s	- set non-sharable file attribute
-s	- reset non-sharable file attribute
d	- set directory attribute (not recommended)
-d	- reset directory attribute (if dir. is empty)

NOTE: Because this program calls ATTR (which comes with OS9) this command must be present in memory or in the current execution directory.

Wmatch

The Wmatch command gives a list of what files match a specified wildcard pathname. The syntax is as follows:

Wmatch <pathname>

A list of all files that match <pathname> will be displayed.

The <pathname> can contain wildcard characters. It can also contain a complete pathlist (as in '/d0/sources/asm/*.a' or a simple file name (as in '*.a'). If the complete path name is not given the current directory will be assumed.

Wmatch is useful for checking what files match the wildcard before copying, deleting, or attring.

DIRECTORY STRUCTURE COMMANDS

The directory structure commands allow you to view the directory structure of your OS9 diskettes easily. The DIR command provided with OS9 allows you to view files, but makes the viewing of directories difficult. This causes you to waste much time going through one directory after another looking for files.

With these commands you can view the entire disk's structure in one sweep, without searching through long directory listings.

These commands are especially helpful when using hard disk drives, as the directory structure in hard drives can become VERY complicated.

Otree

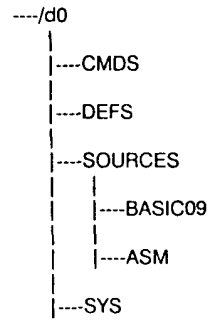
The Otree command shows a graphic display of your disk's directory structure. The syntax is as follows:

Otree <devname>

A display of the directory structure of the disk in the drive specified by <devname> is given.

This allows you to view, graphically, how your disk's directories and sub-directories are organized. The display creates a new 'branch' for every sub-directory from the previous directory.

Example:



Dtree

The Dtree command gives a graphic display of a directory's sub-directories. The syntax is as follows:

Dtree <pathname> [-f]

A display of the directory's structure is given. If the '-f' option is specified, a list of files within the directory will be given.

The output of this command is very similar to Otree. The only difference is that the entire directory structure of a disk need not be viewed. The files can also be listed with the '-f' option.

Example: Otree /d0/SOURCES

```
----/D0/SOURCES
    |
    |----BASIC09
    |
    |----ASM
```

Example: Otree /d0/SOURCES -F

```
----/D0/SOURCES
    | BASIC09
    | ASM
    |
    |----BASIC09
    |    | PROG1
    |    | PROG2
    |    | PROG3
    |
    |----ASM
    |    | PROG1.A
    |    | PROG2.A
    |    | PROG3.A
```

NOTE: Dtree is much more versatile than Otree, but Otree is much more compact and may run faster. It is recommended that Otree be used with floppy systems, and Dtree be used with hard disks (as hard disk directory structures tend to be very large).

Dirsort

The Dirsort command allows you to find files easily by putting them in sorted order. The syntax is as follows:

Dirsort [<pathname>] [-?][-r][-d]

The directory specified by <pathname> will be sorted. If '-?' is specified, no directory will be sorted and the syntax will be listed. if '-r' is specified, the directory will be sorted in reverse order. If '-d' is specified, all directory names will be capitalized, and all file names will be lowercased.

COMMAND FILE COMMANDS

Command files are files that contain a list of commands to be executed one after another. This is useful when more that one operation has to be done to run an application.

Command file commands are commands that give you more versatility in writing command files. This allows you to create a more friendly user environment on your computer by having nice command files run everything.

Pause

The Pause command pauses the computer until a key is pressed. The syntax is as follows:

Pause ["message"] </1

The specified message will be displayed to the screen and the program will wait for a key to be pressed. The brackets ([]) around "message" show that the message is optional. If no message is specified, "Press any key to continue..." will be displayed.

NOTE: The </1 is necessary for the program to take it's input from the keyboard instead of the command file. If this command is used without a command file, the </1 is not needed.

Goto

The Goto command allows you to create loops in a command file. The syntax is as follows:

Goto <tag>

Control will be transferred to the line that contains <tag>.

<tag> specifies a line that begins with an asterisk (*), and is followed by any word. The asterisk (*) causes that line to be ignored during execution so that an error won't be generated. The following is an example that will continuously display the current directory.

Example:

```
* loop
dir e
Goto loop
```

NOTE: Upper and Lower case are considered different. That is 'LOOP' is not the same as 'loop'.

PIPE FILTER COMMANDS

Pipe filter commands are used to reformat the output of other commands to a way that is preferred by you. This is useful for file conversion etc. These commands can be used either as pipe filters or simple filters. To use them as pipes use the following format:

(command) ! (pipefilter)

Example:

Dir ! upcase

This sends the output of the <command> through <pipefilter>. For more information on pipes, see your OS9 manual.

To use a pipefilter command as a simple filter, use the following format:

(pipefilter) <filename

Example:

Upcase </d0/sources/prog1

This sends the specified file through the filter and then out to you. Output can also be re-directed to another file or device using the greater than (>) sign.

Upcase

The Upcase filter takes all characters and converts them to upper case. This is useful when you need to filter out lower case characters.

To change the output of a command to uppercase use the following format:

(command) ! Upcase

Example:

dir ! Upcase

To list a file in uppercase use the following format:

Upcase <(filename)

To convert a file to uppercase use the following format:

Upcase <(inputfile) >(outputfile)

Locase

The Locase filter takes all characters and converts them to lower case. This is useful when you need to filter out upper case characters.

The use of Locase is exactly the same as that of Upcase. For more information, see Upcase.

Dislex

The Dislex filter takes all lines and reverses them. This is useful in encrypting files and just for having fun.

To reverse the output of an OS9 command, use the following format:

(command) ! dislex

Example:

dir ! dislex

To view the reverse of output of a file use the following format:

Dislex <(filename)

To convert a file into reversed order use the following format:

Dislex <(inputfile) >(outputfile)

UTILITY COMMANDS

The utility commands are commands that can be useful in the day-to-day life of a computer user. I find myself using and relying on these tools constantly from day-to-day.

Ascii

The Ascii command gives the ascii value of a character. This is useful when programming. The syntax is as follows:

Ascii <char> [<char>...]

The ascii values (in decimal) will be returned as follows:

The ascii value of <char> is <num>
.
.
.

Note that more than one value can be gotten in a single command.

Convert

The convert command will convert a number from one base to any other base. The syntax is as follows:

Convert <number> <frombase> <tobase>

<number> will be convert from <frombase> notation into <tobase> notation. This can be especially useful in programming when different bases are used.

Examples:

Convert FF 16 10

FF converted from base 2 to base 10 is 255

Convert 11111111 2 10

11111111 converted from base 2 to base 10 is 255

Devname

Devname is used to get the device name associated with a device or file. The syntax is as follows:

Devname [<name>]

The device associated with <name> will be returned.

This command has several uses. Note that <name> is optional (enclosed in brackets). If no name is specified, the current input device is assumed. This allows you to find out what device is associated with your terminal. For Level II users, this allows you to find out what window you are currently working in.

This command can also be used to determine what disk device is currently active by using the following format:

Devname .

The dot ('.') specifies the current directory as the device in question. This is much faster than using PWD, but does not tell you what directory is active.

Calendar

Calendar is a versatile, interactive calendar program that allows you to schedule events, and reminds you of coming events. The syntax is as follows:

Calendar [-s][-r][-?]

An event can either be scheduled (-s), or read (-r).

If you choose to schedule an event (-s), you will be prompted to enter the date of the event, the time of the event, the reminding period, and up to 80 lines of text.

The time of the event can be specified as 00:00:00, this causes the time not to be displayed when the event is displayed.

The reminding period is the number of days before the event to remind you that the event is coming.

If you choose to read an event (-r), you will be given a list of events scheduled for that day. The output format is as follows:

```
Messages for today      Monday January 1, 1900
-------------------------------------------------
<message text>
.
.
.

Reminders for today     Monday January 1, 1900
-------------------------------------------------
ON: Tuesday January 1, 1901, 365 days from today
<message text>
.
.
.
```

The -? option gives a list of the proper syntax.

Browse

The Browse command allows you to look through a file forward and backward very easily. The syntax is as follows:

Browse <filename>

The <filename> should be an OS9 text file. The first page of the file will be displayed on the screen. You can scroll up and down through the file with the arrow keys. If you want to jump a page at a time you can use the shift and arrow keys.

Mmap

The Mmap command displays a map of the memory in the computer. The syntax is as follows:

Mmap

The output from this command should look like this:

```
OS9 Level II Memory Map
Copyright (C) 1987
By Keith Alphonso
-----------------------
  0123456789ABCDEF

0 XXXXXXXXXXXXXXXX
1 XXXXXXXXXOOOXXXX
2 XXXXOOOOOOOOOOOO
3 OOOOOOOOOOOOOOOO
```

An 'X' marks an occupied block, while an 'O' marks a free block. This is a much better way of viewing memory than the Mfree command that comes with OS9 as it is much easier to read.

COLOR MANIPULATION COMMANDS

The color manipulation commands allow you to control the color of your screen. This is useful as constantly looking up the proper control codes can be tedious.

Fcolor

The Fcolor command changes the color of the foreground in the current window. The syntax is as follows:

Fcolor <color>

The <color> you specify should be a number from 1 to 16. The actual color will depend on what type of screen you have and what palettes are currently in effect (see palette). Some screen modes do not support all 16 colors.

Bcolor

The Bcolor command changes the color of the background in the current window. The format for this command is exactly as that of the Fcolor command (above).

Border

The Border command changes the color of the border in the current window. The format for this command is exactly the same as that of Bcolor and Fcolor (above).

Palette

The Palette command changes the palette of a specified color. The syntax is as follows:

Palette <color> <palette> [<color> <palette>...]

The color can be any of the 16 color 'slots', and the palette can be any of the 64 possible colors.

As you know, the COCO III is capable of display 64 different colors. The problem is that it can show at most 16 at one time. Because of this you are allowed to place any of the 64 possible colors into any of the 16 color slots. This command makes the color placement easy.

Notice that more than one palette can be specified at a time. This allows you to set your entire color environment in one command line.

WINDOW HANDLING COMMANDS

The window handling commands give you greater flexibility in the way your windows are set up in OS9. It also gives you the ability to change your window setup easily.

Wconfig

The Wconfig command allows you to change the configuration parameters of the current window. The syntax is as follows:

Wconfig <sty> <cpx> <cpy> <szx> <szy> <fgn> <bgn> <brd>

<sty> = Screen type (as listed in OS9 manual)
<cpx> = X coordinate of upper left hand corner.
<cpy> = Y coordinate of upper left hand corner.
<szx> = Width of the window.
<szy> = Length of the window.
<fgn> = Foreground color.
<bgn> = Background color.
<brd> = Border color.

The syntax for the Wconfig command is very similar to the syntax of the Wcreate command that comes with OS9. This is so that the command will be easier to use. The difference between Wcreate and Wconfig is that Wconfig works on the current window. This means that you can easily change the parameters of the current window without having to open a completely new window.

Window

The Window command creates an overlay window on the current screen, complete with a border and a title. The syntax is as follows:

Window <cpx> <cpy> <szx> <szy> <fgn> <bgn> <brd> <title>

<cpx> = X coordinate of upper left hand corner.
<cpy> = Y coordinate of upper left hand corner.

<szx> = Width of window.
<szy> = Length of window.
<fgn> = Foreground color.
<bgn> = Background color.
<brd> = Border color.
<title> = Window title text.

The overlay window is created at the coordinates specified, with a border around it and the window title centered in the top window border. This command allows you to create very impressive looking displays with command files.

Wend

The Wend command ends a window created with the Window command. The syntax is as follows:

Wend

The current overlay window will be closed, and whatever text was underneath it will be restored. If there is no current overlay window, an error 195 will result.