Unixcorn

Version 2.0 - January 16th, 1995

What is Unixcorn?

Unixcorn is a command line interface for Windows.

"Oh, no," you say, "not **another** command line interface. What makes this one special?"

Unixcorn is special because it can run from a DOS prompt. This has several advantages:

- All built-in commands of DOS shells such as Norton Commander and 4DOS are still available, including far better batch file support than I could write.

- The Unixcorn program is small, since it doesn't have to duplicate all the common DOS commands such as `DEL`, `DIR`, etc.

- Output of DOS commands can be piped to the clipboard, and the current clipboard contents can be used as input to DOS commands.

Unixcorn also has a wealth of analysis and information functions, to help you visualize how Windows and the new applications that run under it eat so much memory. And it won't change a single line of your `win.ini` or `system.ini` files or place a single file in your Windows directory.

With version 1.2, Unixcorn includes a Windows-based command line interface and support for batch files. This allows you to add valuable functionality to programs like Program Manager, Backmenu, Dropper, Appdesk, and other desktop shells. For example, you could click on an icon in Program Manager and bring up a detailed report on Windows' memory usage, or use a batch file to make sure that Notepad always starts in the bottom corner of your screen.

With version 2.0, Unixcorn includes a small TSR program which improves response time and the ability to store output in an environment variable for easier batch file processing.

Unixcorn requires Windows 3.1 running in 386 Enhanced mode.

See Also

How do I Use Unixcorn?, Unixcorn is Shareware (How to Register)

How Do I Use Unixcorn?

To support its unique interface, Unixcorn requires two parts. The first part, `UNIXCORN.EXE`, is a Windows program that should be added to the load= line in `win.ini` or placed in the startup group in Program Manager. It's safe to leave Unixcorn running all the time, since it consumes less than 80 KB of memory and less than 2% of free system resources. The second part takes a command and passes it to Unixcorn, and prints any output from the command. This part is either `UC.EXE`, a DOS program, or `UCWIN.EXE`, a Windows program.

Installing Unixcorn

After unpacking the Unixcorn archive or inserting the Unixcorn disk, you should see the following files:

`readme.txt`	Late-breaking news on Unixcorn
`uc.exe`	DOS interface to Unixcorn
`ucwin.exe`	Windows interface to Unixcorn
`unixcorn.exe`	Unixcorn main program
`unixctsr.com`	UnixcTSR interprocess communication TSR for Unixcorn
`unixcorn.hlp`	Help file for Unixcorn

Place all the Unixcorn files in the same directory. Make sure the directory is on your path; otherwise, DOS won't be able to find `UC.EXE` to run commands.

To make an icon for Unixcorn in Program Manager, open the Startup group and select **New** from the **File** menu. Create a new program item. Type "Unixcorn" for the description. Then click on the Command line box and type `UNIXCORN.EXE`. Hit enter, and you should see the icon appear in Program Manager.

UnixcTSR is a small program run when your system boots which lets Unixcorn give more information to its DOS interface. Using UnixcTSR is optional, but will improve performance. To use UnixcTSR, put a line near the end of your `autoexec.bat` file like the following:

```
unixctsr
```

You can edit your autoexec.bat file by typing "`edit c:\autoexec.bat`" from any DOS prompt. If you start Windows from your autoexec.bat, make sure the line which runs UnixcTSR is before the line which starts Windows (which is usually "`win`" on the last line of your `autoexec.bat`).

Rest assured that Unixcorn won't touch a single file in your Windows directory (not even `win.ini`) unless you explicitly tell it to.

See Also

Uninstalling Unixcorn, Making Unixcorn the Shell

Using Unixcorn from DOS

You use the UC program to communicate with Unixcorn from DOS.

Syntax **UC [-p | -np] [-q] [-i] [-nt] [-ne | -e*varname*] *command***

Switches

-p Pause the output every screen full. This switch takes precedence over the pause state set with the PAUSE command.

-np Don't pause the output every screen full. This switch takes precedence over the pause state set with the PAUSE command.

-q Don't print "Waiting for response..." while waiting for a response from Unixcorn.

-i Ignore output from Unixcorn and return to the DOS prompt immediately. Use with care (especially in batch files).

-nt Don't use the UnixcTSR interface, even if it's available.

-ne Don't set the `UCOUTPUT` environment variable.

-e*varname* Set the environment variable ***varname*** instead of `UCOUTPUT`.

Parameters

command The command to run.

Quick Start If Unixcorn isn't running, click on the icon you just created in Program Manager. Then start a MS-DOS prompt (there should be an icon for it in the Main group in Program Manager). You can toggle the DOS prompt between full-screen and windowed mode using Alt+Enter.

At the DOS prompt, type "`UC ?`". After a second or so, a list of commands should appear on the screen. Now type "`UC TASKS`". You should see the currently running tasks, and maybe a few that don't usually appear in Task Manager (Unixcorn shows you hidden tasks as well). Congratulations, you're now using Unixcorn!

You can get full on-line help by typing "`UC HELP`".

To quit both parts of Unixcorn, type "`UC QUIT`".

UC and Environment Variables

By default, UC will create an environment variable called `%UCOUTPUT` when it exits. Most commands will place some sort of output in this variable (the exact output depends on the command). This allows you to use the output of UC in a batch file. If a command which would normally set the environment is unsuccessful (for example, trying to minimize a nonexistant window), the `%UCOUTPUT` variable will not be set.

Example

```
UC WHAT SCREEN
ECHO Your screen size is %UCOUTPUT
```

If you want to write to a variable name besides `%UCOUTPUT`, use the -e option to specify a new variable name. The name must be shorter than 31 characters.

Example

```
UC -eSCRRES WHAT SCREEN
ECHO Your screen size is %SCRRES
```

In order for Unixcorn to store the environment variable, you need to have enough free environment space. If you can't get Unixcorn to set `%UCOUTPUT`, try typing

```
UC ISET system.ini [NonWindowsApp] CommandEnvSize=2048
```

from any DOS prompt and then restarting your system. This tells Windows to reserve more space for the environment. In any case, due to limitations of MS-DOS, Unixcorn is unable to store more than 127 characters of output to any environment variable.

UC and MS-DOS Errorlevel

The UC program will set the MS-DOS errorlevel to indicate the result of the command. This lets you build MS-DOS batch files which respond interactively to your needs via Windows.

Errorlevel	Meaning
0	Command completed successfully. If command was ECHO, user clicked on Ok or Yes button.
1	Command failed. If command was ECHO, user clicked on Cancel or No button.
2	Unknown command.
3	Error processing batch file.
4	Timeout - couldn't get a response from Unixcorn. Make sure Unixcorn is running.

See Also

Using Unixcorn from Windows, UnixcTSR, Command Groups

UnixcTSR

UnixcTSR is a small (176 bytes when loaded) Terminate-and-Stay-Resident (TSR) program used by Unixcorn and UC to speed communicate with each other. If UnixcTSR is loaded, UC can talk to Unixcorn about twice as fast as without it.

If you plan on using RUN -WAIT in batch files to pause the batch file until a program finishes, UnixcTSR will allow UC to keep track of the program without using the clipboard to communicate with Unixcorn. If you experience problems copying and pasting while a program you started with RUN -WAIT is running, installing UnixcTSR will solve those problems.

UnixcTSR communicates using the alternate multiplex interrupt specification,a series of strict rules for TSR programs to follow which minimizes the chance they will conflict with each other.

You don't have to use UnixcTSR if you don't want to; all the Unixcorn commands will work without it.

See Also

Using Unixcorn from DOS, Installing Unixcorn, Uninstalling Unixcorn

Using Unixcorn from Windows

From the Unixcorn Icon (Unixcorn Command Line)

You may run Unixcorn commands from Windows by bringing up Unixcorn's system menu. Click on the Unixcorn icon, then select "Command line..." from the menu. You can also bring up the command line by double-clicking the Unixcorn icon. You can then type in any Unixcorn command directly (for example, "`TASKS -A`" or "`CLOSE *`"). You can also click on the down arrow to select one of the last eight commands you've typed into the Unixcorn Command Line.

If the command produces output, it will be displayed in a UcWin window.

You can get full on-line help by selecting the "Help..." item in Unixcorn's system menu.

From Program Manager or Some Other Program

You may also use the UcWin program to run commands or batch files from Program Manager, a menuing program like Backmenu, or a button bar like Dropper, AppBar, or AppDock. If Unixcorn isn't running, UcWin will start it and then process the command.

From This Help File

Click the **UcWin** button on the button bar at the top of this window to display the Unixcorn Command Line. This lets you try out commands as you read about them.

See Also

Using Unixcorn from DOS, UcWin, Making Unixcorn the Shell

UcWin

The UcWin program provides an interface to Unixcorn that doesn't require starting an MS-DOS prompt. It functions similar to Notepad or Clipboard Viewer, so if you're familiar with those programs you should also be able to use UcWin.

Syntax **UCWIN *command***

Parameters ***command*** The command to run. If the command is PROMPT, UcWin will display the Unixcorn Command Line dialog and let you type a command.

If a command run from Windows produces output, it will be displayed in a UcWin window. You may have more than one UcWin window open at a time, and may start another UcWin window at any time by selecting "New Window" from the File menu.

*You can start UcWin now by clicking on the **UcWin** button on the button bar at the top of this window.*

You can select and copy text from the UcWin window; however, you can't edit the output. You can use the following hotkeys in UcWin:

Key	Meaning
Ctrl+A	Select the entire output
Ctrl+C, Ctrl+Ins	Copy the selected text to the clipboard
Ctrl+N	Start another UcWin window
Ctrl+P	Rerun the previous command
Tab	Switch between the output and command line

You may type Unixcorn commands into the combo box at the bottom of the UcWin window (for example, "`TASKS -A`" or "`CLOSE *`"). You can also click on the down arrow to select one of the last eight commands you've typed in. Type Enter to run the command you've typed in or selected.

Running UcWin Directly

If you select **Run** from the **File** menu of Program Manager or File Manager, you can enter any Unixcorn command by preceding it with `UCWIN`. (For example, to run the TASKS command you would enter "`UCWIN TASKS`".)

If the command does not produce any output, UcWin will quietly exit after running the command.

Creating Program Manager Icons for Commands

To make an icon for a command, select **New** from the **File** menu of Program Manager. Create a new program item. In the Command Line part of the Program Item Properties dialog, enter `UCWIN` followed by whatever command you want to run (for example, "`UCWIN STICK *`"). In the Description part, type in a description that matches the command (for example "Pick a Window to Stick"). Then press enter. You can now click on the icon to run the command.

See Also

Using Unixcorn from Windows, Command Groups

Making Unixcorn the Shell

The shell program is responsible for processing the load= and run= lines in `win.ini`, and for asking your permission to exit Windows. The shell program is normally Program Manager.

You can use Unixcorn as the shell program for Windows. Since Unixcorn uses fewer resources and memory than Program Manager, this can free up more memory for running your applications. In addition, Unixcorn starts much faster than Program Manager, since it doesn't have to load all those icons.

To use Unixcorn as the shell, execute the following Unixcorn command (see Using Unixcorn from Windows):

```
ISET system.ini [boot] shell=unixcorn.exe
```

The next time you start Windows, Unixcorn will be the shell. To change back to Program Manager, execute the following Unixcorn command:

```
ISET system.ini [boot] shell=progman.exe
```

See Also

Using Unixcorn from Windows, Installing Unixcorn

Creating Aliases and Menu Items for Common Commands

If you type a command frequently, you can set up an *alias* for that command. For example, instead of typing "`UC PLAY C:\WIN3\MYSOUND.WAV`", you could define an alias so you would only have to type "`UC PMS`". You set and show aliases with the ALIAS command. Aliases are not case sensitive, so "`UC pMs`" will also work.

Unixcorn also has ten slots on its system menu which you can set with the USERMENU command. This lets you easily access the commands you use most, without having to open up a dialog box or DOS prompt. Once you have added a menu item, you can run that command by simply selecting the menu item from Unixcorn's system menu.

Aliases and user menu items have limits. They can only take the place of Unixcorn commands, not DOS commands (if you want to alias DOS commands, I suggest getting a copy of JPSoft's excellent 4DOS command.com replacement). So you could define an alias for "`TASKS -A`", but not for "`UC TASKS -A > TASKS.TXT`". Aliases and user menu items are also limited to 256 characters in length. There is no practical limit on the number of aliases you may have (you could have over 100 256-character aliases and still have room for more), but you are limited to ten user menu items.

See Also

Alias and Menu Commands

Using Unixcorn Batch Files

If you have a series of Unixcorn commands you want to run, you can create a batch file containing all of the commands. Then you can use the BATCH command to run all of the commands at once. This is particularly useful at Windows startup, when you could use a batch file to start several applications and position them on the screen the way you like.

Batch files must contain no more than one command per line. Unixcorn will ignore lines beginning with semicolons (**;**), so you can use them to create comments.

An example batch file is shown below.

```
; startup.ucb - batch file run on startup
;
; Run notepad with list of things to do
run notepad c:\personal\ttd.txt
; Size it and put it in the upper right corner
size 400,500 Notepad
pos -1,0 Notepad
; Change its title
title 'Things to Do' Notepad
;
; Load After Dark
load c:\win3\ad.exe
;
; Keep Unixcorn's icon visible
stick Unixcorn
```

This batch file runs Notepad with a list of things to do, changing its title to "Things to Do" and positioning it in the upper right corner of the screen. It then loads After Dark, and sticks Unixcorn's icon to the monitor glass so it won't be hidden by other windows.

You can't use the following commands in a batch file

BATCH
COPY
CUT
PASTE
PROMPT

To run a batch file at startup

When Unixcorn is started, it looks for a file called `startup.ucb` in its directory. Any Unixcorn commands in this file will be run every time Unixcorn is started.

See Also

BATCH

Entering A Serial Number to Register Unixcorn

When you register Unixcorn, you will get a serial number which will remove the shareware screen on startup. For standard registrations, this serial number will be printed on your Unixcorn disk. To enter this serial number into Unixcorn, select the "Enter Serial Number..." option from Unixcorn's system menu. You will be presented with a dialog box with spaces for your name and serial number. Type your name and serial number **exactly** as they appear on the Unixcorn disk, then click the Ok button.

You should see a dialog box informing you that registration was successful. "(UNREGISTERED)" will disappear from the Unixcorn icon title. The about box for Unixcorn will now show your name and serial number. The "Hide Icon on Startup" option in the system menu will be enabled.

If Unixcorn prints an error message, you may not have typed your name or serial number exactly. Even changing the capitalization of your name or adding an extra space at the end will make Unixcorn refuse to register. This protects your investment in Unixcorn by making it difficult to forge serial numbers for Unixcorn. If you are sure you've typed everything correctly and it still doesn't work, contact me and I'll send you a new serial number.

If you have registered Unixcorn electronically (as a student or foreign registration), write your serial number down somewhere. If the `unixcorn.ini` file gets corrupted or deleted by some other program, you will need to re-enter your serial number.

See Also

Contacting the Author, Unixcorn is Shareware (How to Register)

Hiding the Unixcorn Icon (registered users only)

Hiding the Icon

If you have registered Unixcorn, you will be able to hide the Unixcorn icon to keep your desktop cleaner. (If you haven't, the icon provides a subtle reminder - much less annoying than disabled features or nag screens that pop up every few minutes.)

You can hide the Unixcorn icon for the current session by popping up the Command Line (see Using Unixcorn from Windows) and typing in "`HIDE Unixcorn`".

To hide the Unixcorn icon every time you run Unixcorn, select the "Hide Icon on Startup" option from Unixcorn's system menu. Several seconds later, the icon will disappear (this gives you time to change your mind). The next time Unixcorn is started, it will remember to hide its icon after a few seconds.

Reshowing the Icon

To reshow the Unixcorn icon for the rest of the current session, run Unixcorn a second time. The second copy will show the original copy's icon, then exit.

To keep the icon from disappearing on startup, uncheck the "Hide Icon on Startup" item on Unixcorn's system menu. You can do this on startup before the Unixcorn icon disappears, or you can run Unixcorn a second time to reshow the icon.

See Also

Unixcorn is Shareware (How to Register), HIDE

Uninstalling Unixcorn

If you made Unixcorn the shell, you should change the shell back to Program Manager before uninstalling Unixcorn.

Uninstalling Unixcorn is simple - just delete the files you installed in the first place, and `unixcorn.ini` if it was created (it will be in the same directory as `unixcorn.exe`). If you created a Program Manager icon for Unixcorn, delete that too. Unixcorn doesn't make any changes to your `win.ini` or `system.ini` files, and doesn't place any files in your Windows directory, so you don't have to muck around with those.

If you started UnixcTSR in your `autoexec.bat` file, delete the line in `autoexec.bat` where you started it.

If you created the Unixcorn startup batch file `startup.ucb`, delete that too.

See Also

Installing Unixcorn, Making Unixcorn the Shell

Unixcorn is Shareware (How to Register)

Unixcorn is shareware, and has never been "public domain" or "free" software. The Unixcorn source, executable, and documentation are Copyrighted ©1995 by Randall Spangler. You are allowed a two-week trial period to determine if Unixcorn is worth the registration fee. At the end of those two weeks, you must either register Unixcorn or stop using it. Registering Unixcorn entitles you to the following benefits:

- A serial number which will disable the shareware screens in current and future versions of Unixcorn
- The ability to hide the Unixcorn icon
- Free updates via E-mail
- Support direct from the author via E-mail
- A clean conscience

Depending on your status, you can register Unixcorn one of four ways:

STANDARD REGISTRATION

I believe that the shareware concept works most effectively when the registration fee is reasonable. To support this idea, the registration fee for Unixcorn is only ten dollars ($10). This entitles you to the above benefits, and I will mail the registered version to you on a 3.5" DS/DD disk via US mail (to addresses in the US only).

BUDGET REGISTRATION:

If you are a full-time student, you can register Unixcorn for only five dollars ($5). I'm a starving graduate student too; I understand the financial difficulties many students face. This entitles you to the above benefits, and I will E-mail the registered version to you instead of mailing you a disk. In order to be eligible for student registration, you **MUST** have access to E-mail.

FOREIGN REGISTRATION:

If you live outside of the US and do not have a US mailing address, you are eligible for foreign registration. This is because it's a real hassle to exchange currency (for an inexpensive program such as Unixcorn, you'd spend more converting your currency to US dollars and sending me an international money order than you would on the program. Instead of money, send me a postcard (yes, only a postcard) from your country, and I'll E-mail the registered version to you instead of mailing you a disk. In order to be eligible for foreign registration, you **MUST** have access to E-mail. (If you really want me to mail you a disk, contact me and we'll work out a reasonable registration fee which will cover my postage costs.) Please print or type your E-mail address on your postcard so that even a nearsighted graduate student can read it, and send me a filled out registration form via E-mail if you don't enclose it with the postcard.

CORPORATE/BULK REGISTRATION:

If you are purchasing 10 or more copies of Unixcorn for use in a corporate or network setting, or wish to bundle Unixcorn with a program you are selling, please contact me for significantly discounted rates.

All registration fees should be in the form of a check or money order in US currency. Do not send me cash or stamps; I cannot be responsible if they are lost or stolen in the mail. Please print out and fill in the supplied registration form. Mail it and your registration fee, or (for foreign registration only) a postcard to me at:

Randall R. Spangler
Caltech 116-81
Pasadena, CA 91125

Please allow 2-6 weeks for your disk/datafile to show up. (I may be on the verge of a new release, and I want to wait a week or two and send you the new release instead of the current one. Then again, I may be doing real work for a change...)

These registration offers will remain in effect until December 31st, 1995. After that date, I reserve the right to change the registration offers and costs. If you want to register Unixcorn after that date, please contact me for updated prices.

See Also

Registration Form for Unixcorn, Contacting the Author, Entering A Serial Number to Register Unixcorn

Registration Form for Unixcorn

(for foreign registration, just send me a postcard and E-mail this form)

Please mail this form and your check or money order (do not send cash) to:

Randall Spangler
Caltech 116-81
Pasadena, CA 91125

You can print this form by selecting **Print Topic** from the **File** menu, or you can copy it to the clipboard with **Copy** from the **Edit** menu.

See Also

Contacting the Author, Unixcorn is Shareware (How to Register)

```
-------------------------------------------------------------------------
Unixcorn version:   2.0            Date: _____________________
Name: ____________________________________________________________________
Address line 1: __________________________________________________________
Address line 2: __________________________________________________________
City: _______________ State: __________ Zip: ______ Country: ___________
E-mail address: __________________________________________________________
Type of registration:   [ ] Normal ($10)
                        [ ] Student ($5)
                        [ ] Foreign (postcard)
      If student, what college/university: ______________________________

The following information is not required, but it will help me determine what
features I can add to future versions of Unixcorn:

Computer speed: _____ MHz, type: ____________________
How much memory do you have: ____________

What windowing system do you run:                  version: __________
      [ ] Windows   [ ] Windows NT   [ ] OS/2   [ ] other:_____________

What networking software do you use: _____________________________________
What type of video card do you have: _____________________________________

Where did you find Unixcorn: _____________________________________________
      (if it was a BBS, a phone number would be nice too)
Questions, comments, etc:
```

Distributing Unixcorn

You may copy and distribute this shareware version of Unixcorn via any means, electronic or disk, subject to the following conditions:

- The shareware version of Unixcorn may not be rented or sold, nor bundled with a product that is sold or offered as an incentive to buy a product that is sold, without the prior written permission of the author. Such permission is usually granted. This includes distribution via CD-ROMs.
- Unixcorn must be distributed with all files intact with no modifications. You may repack the archive into a different format (ZIP, ZOO, ARJ, etc.) so long as all the original files contained in the archive are intact.

If you maintain a BBS or ftp site and wish to offer Unixcorn for downloading, please contact me. This allows you to be informed of updates to Unixcorn as they are released. It also allows me to let other people know where to find updated versions.

If you are unable to find Unixcorn on an ftp site or BBS, you may obtain the most recent shareware version by sending me a check for five dollars ($5). I will then mail the program to any US address on a DS/DD 3.5" disk. If you live outside the US, it may cost more to mail a disk to you - contact me for information. (Remember that if you have registered Unixcorn, your serial number will remove the registration reminders from any shareware copy of Unixcorn, so you can easily upgrade this way.)

You may not distribute registered copies of Unixcorn or any serial numbers.

See Also

Contacting the Author

Disclaimer and Warranty:

DISCLAMER:

Windows is not the most stable operating environment in the world. I recommend that you save your work in other applications frequently (heck, this is good advice even if you're not running Unixcorn). It is possible to crash Windows or lose unsaved work with Unixcorn if you're not careful (see in particular the KILL , NUKE, UNLOAD, and ISET commands).

LIMITED WARRANTY:

Unixcorn is provided on an "as is" basis without warranty of any kind, expressed or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Use this program at your own risk. I take no responsibility for any damages of any kind resulting in any way (indirect, consequential, special, whatever) from your use of this software. Laws in your state may vary. Besides, I'm a starving graduate student with nothing for you to take.

Answers to Common Questions

Q: Everything scrolls off the screen so fast. How can I slow it down?
A1: Turn on pausing or use the **-p** option to UC.
A2: Pipe the output of Unixcorn to `MORE`. `MORE` is a program which comes with MS-DOS, which pauses after each screen full of output.

Example `UC -P MODULES`

See Also

Using Unixcorn from DOS, PAUSE

Q: I can't use Unixcorn to exit or restart Windows from a DOS prompt. I get a message box which says "Application still active. Quit the application before exiting Windows."
A: In order to exit Windows from a DOS prompt, the PIF file for that DOS prompt must have the "Allow close when active" box checked. This box is in the Advanced settings dialog in PIF editor.

Q: I'm running a MS-DOS batch file in a DOS prompt, and when Windows switches away from the DOS prompt (for example, if another Windows program is started) the batch file pauses.
A1: Make sure the DOS prompt is set to run in the background using PIF editor.
A2: You can set a currently running program to run in the background by choosing the Settings... option from the DOS prompt's system menu (these settings only apply to the currently running program - they aren't saved to its PIF file).

Q: UC keeps giving up with the error message "No response from Unixcorn"
A1: Make sure Unixcorn is actually running (running it a second time will make sure - it's smart enough to keep only one copy in memory).
A2: The command may be taking longer than UC expects. Install the UnixcTSR before Windows starts; this will let UC know when Unixcorn has started running a command so it won't time out.
A3: UnixcTSR has been installed before Windows started, but isn't able to help communicate between UC and Unixcorn. Try running UC with the **-nt** option to use the old interface to Unixcorn. If that solved your problem, UnixcTSR isn't compatible with your system. Remove it from your `autoexec.bat` file, so the next time you reboot your system Unixcorn will start using the old interface.

See Also

Using Unixcorn from DOS, UnixcTSR, Installing Unixcorn

Q: I registered Unxicorn and typed in my serial number a few weeks ago. Now Unixcorn is displaying the shareware screen again. What gives?
A: Unixcorn stores your name and serial number in `unixcorn.ini`. If this file is deleted or corrupted, Unixcorn will behave like an unregistered copy again. Simply retype your name and serial number and the problem will go away.

See Also

Entering A Serial Number to Register Unixcorn

Known Bugs and Limitations

+ Unixcorn uses the clipboard to communicate between DOS and Windows. This means that every time you use a Unixcorn command from DOS, the clipboard gets emptied. UcWin does not have this limitation, and does not affect the clipboard.

+ Unixcorn can only handle input and output of about 30 KB long (about 400 full lines) If a command (such as HEAPWALK) produces a great deal of output, the end of the output will be lost. In this case, the last line of output will be "`(output truncated)`".

Contacting the Author

Please contact me if you have any comments about Unixcorn, suggestions for features I could add, bug reports (bugs? what bugs?), etc. I can be reached via E-mail (the fastest) or snail mail (US mail - the slowest).

E-mail: rspangle@micro.caltech.edu

US mail: Randall R. Spangler
Caltech 116-81
Pasadena, CA 91125

If you find a bug or undocumented feature, the following information will aid me in tracking down and squashing it (or at least documenting it):

- What version of Unixcorn you are using
- A description of what happened.
- What versions of Windows and DOS you are running
- The contents of your win.ini and system.ini files

I will always respond to E-mail. I will always respond to snail mail too, if you send a self-addressed stamped envelope for me to put my reply in.

Special Thanks To

+ Celeste Sleeper, for designing the spiffy Unixcorn icon, and making meatloaf so I didn't starve while I programmed
+ Shakib A. Otaqui and Piotr Karocki, for suggesting many wonderful commands and improvements.

And all the registered users worldwide who have provided me with inspiration and ideas.

What's New in This Release

Many of these changes were suggested by users on their registration forms. Send yours in today with your ideas!

Version 2.0, January 16th, 1995

+ Fixed WALLPAPER when a full path to the wallpaper file was specified
+ Fixed -I option to UC. Now indicates that UC has ignored output from Unixcorn.
+ Unixcorn automatically searches for and runs `startup.ucb` on startup unless command line option NOSTARTUP is specified
+ Added DEBUG command line option to Unixcorn - lets it do more checking on its interactions with UcWin and UC - Unixcorn's icon title will be in all uppercase to indicate debug mode
+ Added -NT option to UC to skip using UnixcTSR interface (may be necessary for Win95 and WinNT users - use if UC keeps timing out with "No response from Unixcorn" messages and UnixcTSR is loaded before Windows is started)
+ Added -A and -L option to HEAPWALK to print block addresses and low memory only
+ Fixed a bug in reporting free low memory in MEM command
+ Added -WAIT option to RUN command - waits until program finishes before returning
+ Added -H option to HEAPWALK to print block handles
+ Added -I option to ECHO to return immediately (without waiting for user to click 'ok').
+ Added support for UnixcTSR - if loaded, uses this to increase the communication speed between Windows and DOS. Using UnixcTSR also fixes the occasional problem Unixcorn has of switching away from a DOS box which has the clipboard open.
+ Specifying a windowid of "+" will match the active window
+ Fixed serial number checking for versions greater than 1.3
+ The busy cursor is animated, so it's more obvious something is happening during long commands
+ Windows that are minimized and part of the Windows 95 taskbar are reported as so
+ Added -F option to force UNLOAD or NUKE without confirmation (use with care!)
+ Fixed erroneous reporting of "No response from Unixcorn" after a RUN -WAIT or ECHO command
+ Added -I option to UC to ignore output from Unixcorn and exit immediately (use with caution)
+ UC writes an environment variable `UCOUTPUT`, so its output can be used more easily in batch files. Added -NE option to not write the environment variable, and -Evarname option to write the environment output to the specified variable name. The value of `UCOUTPUT` depends on the command being run.
+ Added INPUT command to pop up an input box and ask for a value. Result is printed, and stored in `UCOUTPUT`.
+ Added WHAT command to produce various single tidbits of information (screen resolution, Unixcorn version, Windows version, free resources, free memory, etc.)
+ Added "New Window" option on UcWin file menu to start another UcWin window
+ Added Usermenu menu to UcWin so that user menu options can be selected from within UcWin

Version 1.3, June 24th, 1994

+ Added keyword index to help file
+ Added UNLOAD command to unload a module
+ If no windowid or taskid is specified for a command which requires one, lets the user pick a window (the same as if the user entered "`*`" for the windowid or taskid)
+ Pick dialog now tells user what will be done with the picked window
+ UC sets the MS-DOS errorlevel to reflect the result of the command
+ UC ECHO waits for the user to click a button on the popped up message box before returning to MS-DOS.
+ UcWin will start Unixcorn if it isn't running, instead of complaining
+ During long commands, the cursor will change to a busy cursor

+ Added "Previous Command" option to UcWin to rerun the last command
+ Merged command line dialog onto bottom of UcWin window
+ Finally got double-clicking the Unixcorn icon to pop up the command line
+ Better error checking when accessing the clipboard; may stop crashes reported when used at the same time as WordPerfect for Windows
+ Added total DOS/System usage to MEM command
+ Added number of unused handles in MEM command
+ Added WALLPAPER command to set the wallpaper. Can set wallpaper to be centered or tiled. Can use the -N option to set wallpaper to (none).
+ Added ISMODULE, ISTASK, ISWINDOW functions - set errorlevel 0 if the specified handle is a valid window/module/task, or 1 if not
+ Added -OKCANCEL and -YESNO options to ECHO to control the buttons which are displayed
+ Added -m option to WINDOWS command to display the module/program which owns each window. Added -p option to display the window position (default). -c, -m, and -p can be combined to display lots of info. Class info no longer replaces the window title info in -c.
+ WINDOWS prints the active window's title. If a window is stuck (topmost), prints '+' before "Displayed" column
+ Added Help buttons to all dialogs
+ HELP command can now take a keyword (such as a command) to look for. HELP INDEX displays index.
+ Added switches to RUN command:

  -MIN minimize app (same as LOAD command)
  -MAX maximize app
  -HID run app hidden (same as RUNHIDDEN command)
  -HIG load/run app high (same as LH command)
  -IN dir run app in specified directory

  The LOAD,RUNHIDDEN, and LH commands are now outdated, but will be retained for the next few releases for compatibility.
+ Added "New Window" option to file menu for UcWin.

Version 1.2, May 10th, 1994

+ Fixed command line dialog not auto-scrolling
+ Added spiffy on-line help
+ Fixed calculation of low DOS memory in MEM command
+ Improved detection of swapfile sizes and locations in MEM command
+ The Unixcorn icon cannot be hidden in the shareware version
+ Unixcorn no longer accidentally runs a command that was in the clipboard before Unixcorn was started. For sure this time.
+ Unless another window was purposely activated, after a command returns control to the window which was active before Unixcorn.
+ Fixed UPTIME command returning negative time spent in DOS applications.
+ HEAPWALK indicates which items are in low memory; added LOW option to print only items in low memory.
+ CLEAR command works from the Windows command line. CUT, COPY, and PASTE still only work from DOS.
+ Fixed translation of end-of-line between CR and CRLF
+ Added LH command to try to keep a Windows program from using all remaining low memory
+ Added RUNHIDDEN command to run a program hidden
+ Fixed TITLE command if no new title was specified
+ Window names no longer need to be specified exactly, as long as the first part of the window name matches. So you can type "Notepad" instead of "Notepad - (Untitled)".
+ Added accelerator keys for About... and Command Line... in system menu
+ Added COMPACT command to free all discardable memory
+ Fixed crash if input to CUT command was too long
+ Can specify negative coordinates to POS to position a window relative to the lower right hand corner

of the screen.
+ Improved picking windows using the mouse. Now displays a dialog box with the selected window's title and icon.
+ Added CLASSES command to list window classes
+ Added IGET and ISET commands to get and set .INI file entries
+ Output of HELP command is always paused to make it easier to read
+ Added dialog to enter serial number to register Unixcorn and get rid of that pesky shareware screen
+ Added USERMENU command to add user-defined commands to the Unixcorn system menu
+ If output is paused, UC now detects the current screen size and pauses once per screenful instead of once every 24 lines

Version 1.1, April 6th, 1994

+ Fixed PLAY command if the wavefile to be played was on a different drive than the Windows directory.
+ Added "Command line..." option to Unixcorn icon menu.
+ Added HIDE command line option to Unixcorn (available in registered copies only).
+ Added ALIAS command to create and delete aliases.
+ All commands may be abbreviated to their first three letters
+ If "*" is given as a window name, lets user pick a window to act upon.
+ Running Unixcorn a second time now activates the first copy instead of beeping.
+ Unixcorn no longer accidentally runs a command that was in the clipboard before Unixcorn was started.
+ ECHO reactivates the application that was active before the echo command.
+ Added PAUSE command to set whether DOS output is paused every screen full

Version 1.0, March 27th, 1994

+ Initial release

THINGS TO DO IN THIS HELP FILE STILL

- Change Arial to MS Sans Serif and Courier New to Courier
- Add SHOW ME THE EXAMPLE macros to run the Unixcorn commands for each example
- Have boxes after each command for "UC", "UCWIN", "Unixcorn Command Line", like in MS-DOS manual?
- Add section for "other software by the author"

For each new release, update

- Version number at top
- Version number in registration form
- Memory consumed in how do I use unixcorn
- Manifest in installing unixcorn
- Update "Special thanks to" to include people who suggested features for new release
- Make sure copyrights are correct in resource files and in this doc
- Search for "KLUDGE" in source code, make sure unneeded statements are removed
- Make sure all version numbers of UC, UcWin, Unixcorn EXE's display correctly
- Build new key phrase table for help file and rebuild on high compression
- Rebuild all EXE's with optimizations on

Things new in 2.0

Alphabetical List of Commands

The following section lists all available commands in alphabetical order. For any command, unless otherwise noted you may type the entire name (i.e. **MODULES**) or just the first three letters (i.e. **MOD**). If the first three letters don't make a unique command, Unixcorn will tell you which commands start with those letters.

UPTIME
USERMENU
WALLPAPER
WHAT
WINDOWS

See Also

Command groups

Command Groups

The following section lists all available commands, grouped by function. For any command, unless otherwise noted you may type the entire name (i.e. **MODULES**) or just the first three letters (i.e. **MOD**). If the first three letters don't make a unique command, Unixcorn will tell you which commands start with those letters.

See Also

Alphabetical list of commands

Clipboard Control Commands

These commands all affect the clipboard. They provide functionality similar to the Edit menu in Windows applications.

CUT
COPY
PASTE
CLEAR

Application Starting and Task Commands

These commands start, list, and end tasks (applications).

RUN

TASKS
ISTASK

CLOSE
KILL
NUKE

Window Commands

These commands list or modify windows.

WINDOWS
CLASSES
ISWINDOW

RESTORE
MIN
MAX

HIDE
SHOW

ACTIVATE
RAISE
LOWER
STICK
UNSTICK

SIZE
POS

TITLE

ARRICONS

Memory Management Commands

These commands provide information on the ways Windows is currently using memory.

MEM
HEAPWALK
COMPACT

MODULES
ISMODULE
UNLOAD

Alias and Menu Commands

These commands provide easier ways for you to run commands you use frequently.

ALIAS
USERMENU

See Also

Creating Aliases and Menu Items for Common Commands

Exit Commands

These commands exit Unixcorn or Windows.

QUIT

EXIT
RESTART
REBOOT

Other Commands

These commands don't fit into well-defined groups.

IGET
ISET

ECHO
INPUT

PLAY
WALLPAPER

WHAT
UPTIME

BATCH
PROMPT

PAUSE

?
HELP

COMMANDS LISTED ALPHABETICALLY:

?

Lists the available Unixcorn commands, with a short explanation of each one.

Syntax **?**

Note The output of the **?** command is always paused every screen full, regardless of the PAUSE setting.

Example ?

See Also

HELP

ACTIVATE

Switches to the specified window and brings it to the top. This is the same as selecting **Switch To** from Task Manager.

Syntax **ACTIVATE *windowid***

Parameters windowid The window to activate.

Environment Sets `%UCOUTPUT` to the *windowid* of the activated window.

Example

```
ACTIVATE Notepad
```

Switches to Notepad.

See Also

RAISE, RESTORE

ALIAS

Sets and shows Unixcorn aliases.

Syntax **ALIAS [*name*] [*command*]**

Parameters

name The name of the alias to be created or deleted. If not specified, all currently defined aliases are listed.

command The command to be executed in place of the alias. If not specified, the alias with the given name is deleted.

Example

```
ALIAS
t=tasks -a
fool=ECHO Stop playing around you fool!
cw=close *
```

Prints all currently defined aliases.

Example

```
ALIAS mdll MODULES DLL
```

Creates an alias "`mdll`" which will execute the MODULES command "`MODULES DLL`" to list all currently loaded dynamic link libraries.

Example

```
ALIAS mdll
```

Deletes the alias "`mdll`".

See Also

Creating Aliases and Menu Items for Common Commands
USERMENU

ARRICONS

Arranges the icons of all inactive windows at the bottom of the desktop. This is identical to selecting **Arrange Icons** from Task Manager.

Syntax **ARRICONS**

Example

```
ARRICONS
```

See Also

POS

BATCH

Runs a series of Unixcorn commands from a file.

Syntax **BATCH [*batchfile*]**

Parameters ***batchfile*** The text file containing the commands.

Note This command can only be specified on the command line of UC or UcWin. It may not be entered into a Unixcorn Command Line, or be part of a batch file itself.

The batchfile parameter may only be omitted if the command is run from UC; in this case, commands will be taken from standard input.

Example `UC BATCH startup.ucb`

This will run the commands in `startup.ucb`.

See Also

Using Unixcorn Batch Files

CLASSES

Lists the window classes known to Windows; mostly useful for programmers and hackers.

Syntax **CLASSES [-p]**

Switches **-p** Tries to figure out which module is getting the messages from each window class; this can help you track down problems related to window subclassing. If the module can't be identified, Unixcorn prints a ? instead.

Environment Sets `%UCOUTPUT` to the number of classes printed.

Note If the -p option is used, the **CLASSES** command may take a few seconds to complete. The cursor will change to an animated busy indicator. Don't panic and assume Unixcorn has crashed.

Example

```
CLASSES -p
Class                            Module   Wndproc
-------------------------------- -------- --------
Unixcorn                         UNIXCORN UNIXCORN
WOAFontPreview                   WINOLDAP WINOLDAP
WOAWinPreview                    WINOLDAP WINOLDAP
tty                              WINOLDAP WINOLDAP
MDIClient                        USER     USER
ComboBox                         USER     USER
ComboLBox                        USER     USER
ScrollBar                        USER     USER
ListBox                          USER     USER
Edit                             USER     USER
Static                           USER     ?
Button                           USER     USER
```

See Also

MODULES, TASKS, WINDOWS

CLEAR

Deletes the current clipboard contents.

Syntax **CLEAR**

Example `CLEAR`

See Also

COPY, CUT, PASTE

CLOSE

Asks a task to exit. This is the same as selecting Close from the system menu of the application. If there is unsaved work, the application will usually ask if you want to save the changes. This is the most polite way of ending a task.

Syntax **CLOSE *taskid***

Parameters taskid The task to close.

Note Closing the shell task will also exit Windows.

Example

```
CLOSE 13711
```

This will close Microsoft Word, after it asks if you want to save changes to UNIXCORN.DOC.

See Also

KILL, NUKE

COMPACT

Frees all discardable Windows memory. This will increase the amount of free physical memory on your system. If you are having trouble loading a new program, this may clear up the problem.

Syntax **COMPACT**

Environment Sets `%UCOUTPUT` to the amount of free physical memory, in KB.

Example `COMPACT`

Frees all discardable memory.

See Also

MEM, HEAPWALK LRU

COPY

Transparently copies input to the clipboard. This differs from CUT because it also outputs the text. The input is limited to about 30 KB (around 400 full lines of text).

Syntax **COPY**

Note This command only works with UC.

Example `DIR | UC COPY > DIRLIST.TXT`

This will place the output of the `DIR` command on the clipboard, and another copy of the output in the file `DIRLIST.TXT`.

See Also

CLEAR, CUT, PASTE

CUT

Places input to UC on the clipboard. The input is limited to about 30 KB (around 400 full lines of text).

Syntax **CUT**

Note This command only works with UC.

Example

```
DIR | UC CUT
```

This will place the output of the DIR command on the clipboard, where you can paste it into another application.

See Also

CLEAR, COPY, PASTE

ECHO

Pops up a message box with the specified text. A good use for this would be in a MS-DOS batch file running in the background; you could have it pop up a message box to let you know when it's finished.

Syntax **ECHO [-o | -y | -i] *text***

Parameters ***text*** The text to be displayed.

Switches

-o Short for **-okcancel**. Creates a message box with "Ok" and "Cancel" buttons. UC returns errorlevel 0 if "Ok" was pressed, or errorlevel 1 if "Cancel" was pressed.

-y Short for **-yesno**. Creates a message box with "Yes" and "No" buttons. UC returns errorlevel 0 if "Yes" was pressed, or errorlevel 1 if "No" was pressed.

-i Short for **-immediately**. UC returns immediately, without waiting for the user to click "Ok". Good for displaying status messages without slowing down the batch file. Note that Unixcorn will ignore subsequent commands it is sent until the user dismisses the first message box.

Environment Sets `%UCOUTPUT` to 0 if no/cancel pressed or 1 if yes/ok pressed.

Example

```
ECHO Hello, World!
```

Pops up a message box with the text "Hello, World!".

Example

```
REM  Batch file demonstrating ECHO
UC ECHO -yesno Run Norton Disk Doctor?
IF NOT ERRORLEVEL 1 NDD /q C:
```

Runs Norton Disk Doctor only if the user clicks the "Yes" button.

See Also

INPUT, PLAY

EXIT

Exits Windows.

Syntax **EXIT**

Note You will be asked to confirm this command before Unixcorn actually exits Windows.

Example `EXIT`

See Also

QUIT, REBOOT, RESTART

HEAPWALK

Prints the contents of one of Windows' global heaps. These are where Windows keeps track of all memory used by Windows applications. (Memory used by DOS applications is hidden away where Windows apps can't see it.) The heaps are usually hundreds or thousands of items long (see global heap items in the MEM command), so this command often prints only the first thousand or so items.

Syntax **HEAPWALK [-a | -h | -l] [ALL | LRU | FREE | LOW | *moduleid*]**

Parameters You may specify one of the following types of memory to look at:

LRU	All discardable memory. This memory can be temporarily freed with the COMPACT command, since Windows will re-allocate it when needed.
FREE	Unused memory blocks.
ALL	All blocks of memory allocated by Windows. If you don't specify a heap, Unixcorn prints the **ALL** heap.
LOW	Only those items in the **ALL** heap which are occupying low memory.
moduleid	Memory owned by a single module.

Switches

-a	Short for **-addresses**. Prints the memory address of each block.
-h	Short for **-handles**. Prints handle of each block.
-l	Short for **-lowonly**. Prints only low memory blocks in the specified list of items.

Environment Sets `%UCOUTPUT` to the total memory in the selected heap, in bytes.

Example

```
HEAPWALK
CONTENTS OF ALL HEAP:
  Size   Owner Module   Type
     0 L 65535          (first/last object)
 43328 L   279 KERNEL   code
    64 L   967 GDI      (unknown)
  1152 L  5463 MMSYSTEM code
    32    5463 MMSYSTEM data
   320    5455 TIMER    code
   576    5455 TIMER    default data segment
  3680    5367 MVPROAUD code
    32    5367 MVPROAUD code
  3552     279 KERNEL   code
  2208    5367 MVPROAUD default data segment
  2560    5263 MMMIXER  default data segment
```

For each item on the heap, Unixcorn prints its size, the module which owns it, and the type of item. Unixcorn prints an 'L' after the size of each item which is in low memory.

If you're more interested in the memory used by a given module, you can specify the module's name or handle.

Example

```
HEAPWALK NOTEPAD
CONTENTS OF HEAP OWNED BY NOTEPAD:
  Size   Owner Module   Type
```

```
   512 L 14255 NOTEPAD  task data
   672 L 14295 NOTEPAD  module data
    64 L 14295 NOTEPAD  resource: icon
    64 L 14295 NOTEPAD  resource: accelerators
    96 L 14295 NOTEPAD  resource: accelerators
   448 L 14295 NOTEPAD  resource: menu
   544 L 14295 NOTEPAD  resource: string table
   320 L 14295 NOTEPAD  resource: string table
   416 L 14295 NOTEPAD  resource: string table
   320 L 14295 NOTEPAD  resource: icon
  1184 L 14295 NOTEPAD  resource: icon
  1440 L 14295 NOTEPAD  resource: string table
 14880   14295 NOTEPAD  default data segment
 12800   14295 NOTEPAD  code
  2176   14295 NOTEPAD  code

Total size: 35936 bytes
```

Prints only the memory items used by Notepad. Note that there are two different owners for Notepad: 14255 is the task, and 14295 is the module which holds the code for the task.

See Also

COMPACT, MEM, MODULES

HELP

Starts Windows Help with the full on-line documentation for Unixcorn, or searches for help on a particular topic.

Syntax **HELP [*topic*]**

Parameters ***topic*** Any topic you might want help on, such as a command name. Using `INDEX` as a topic will let you search through a list of keywords.

Example `HELP`

Starts Unixcorn help at the table of contents.

Example `HELP MEM`

Starts Unixcorn help at the explanation of the MEM command.

See Also

?

HIDE

Hides a window. Hidden windows do not appear in Task Manager or in the WINDOWS command, but do show up in WINDOWS -h. Hidden windows also cannot be sized or positioned. Aside from that, most hidden programs run normally.

Syntax **HIDE *windowid***

Parameters windowid The window to hide.

Environment Sets `%UCOUTPUT` to the *windowid* of the hidden window.

Example

```
HIDE Unixcorn
```

Hides Unixcorn's icon. (However, in the shareware version the icon will reappear the next time a command is run.)

See Also

RUNHIDDEN, SHOW

IGET

Prints an entry from an .INI file. This is a good way to keep tabs on your Windows configuration.

Syntax **IGET [*file*] *section entry***

Parameters

file The .INI file to read from. If this parameter is left out, Unixcorn assumes you mean `win.ini`.

section The section of the .INI file containing the entry. This must be enclosed in brackets ("`[desktop]`" will work, but "`desktop`" won't).

entry The name of the entry to read.

Environment Sets `%UCOUTPUT` to the value for the entry.

Example

```
IGET [windows] load
c:\win3\afterdrk\ad.exe atikey.exe
```

Prints the contents of the load= line in `win.ini`.

Example

```
IGET clock.ini [clock] position
849,20,969,108
```

Gets the location where Windows' clock appears when run.

See Also

ISET

INPUT

Pops up a message box with the specified text, then waits for the user to type a response. This is a good way to get user input into a batch file. Sets errorlevel 0 if successful, or 1 if the user pressed cancel.

Syntax **INPUT *text***

Parameters ***text*** The text to be displayed above the input box.

Environment Sets `%UCOUTPUT` to the user's input.

Example

```
INPUT What is your name?
```

Pops up a message box with the text "What is your name?" then waits for the user to input something. Then prints what the user typed into the box.

Example

```
REM  Batch file demonstrating INPUT
UC INPUT What is your quest?
IF %UCOUTPUT == HolyGrail UC ECHO Monty Python alert!
```

Pops up the echo box only if the user types in the correct text

See Also

ECHO

ISET

Sets an entry from an .INI file.

Syntax **ISET *file* [*section*] *entry*[=*value*]**

Parameters

file	The .INI file to write to. If this parameter is left out, Unixcorn assumes you mean `win.ini`.
section	The section of the .INI file containing the entry. This must be enclosed in brackets ("`[desktop]`" will work, but "`desktop`" won't).
entry	The name of the entry to write.
value	The new value for the entry. If omitted, removes the entry

Environment Sets `%UCOUTPUT` to the new value for the entry.

Note Be careful with this command - you could conceivably change or delete a setting Windows needs to run.

Example

```
ISET [desktop] wallpaper=c:\win3\frog.bmp
```

Sets the wallpaper which Windows will use in the background the next time it's started.

See Also

IGET

ISMODULE

Sets MS-DOS errorlevel 0 if the moduleid is valid, or 1 if it is invalid. Only useful from UC. This is a good way to fill `%UCOUTPUT` with a moduleid.

Syntax **ISMODULE *moduleid***

Parameters moduleid The moduleid to check.

Environment Sets `%UCOUTPUT` to the *moduleid* of the found module, or 0 if failure.

Example

```
ISMODULE GDI
```

Sets errorlevel 0, since GDI is the name of a module currently loaded into memory.

See Also

ISTASK, ISWINDOW, MODULES

ISTASK

Sets MS-DOS errorlevel 0 if the taskid is valid, or 1 if it is invalid. Only useful from UC. This is a good way to fill `%UCOUTPUT` with a taskid.

Syntax **ISTASK *taskid***

Parameters taskid The taskid to check.

Environment Sets `%UCOUTPUT` to the *taskid* of the found task, or 0 if failure.

Example

```
ISTASK Flooble
```

Sets errorlevel 1, since there is no task called "Flooble" currently running.

See Also

ISMODULE, ISWINDOW, TASKS

ISWINDOW

Sets MS-DOS errorlevel 0 if the windowid is valid, or 1 if it is invalid. Only useful from UC. This is a good way to fill `%UCOUTPUT` with a windowid for subsequent commands.

Syntax **ISWINDOW *windowid***

Parameters windowid The windowid to check.

Environment Sets `%UCOUTPUT` to the *windowid* of the found window, or 0 if failure.

Example

```
ISWINDOW Unixcorn
```

Sets errorlevel 0, since there is currently a window titled "Unixcorn".

See Also

ISMODULE, ISTASK, WINDOWS

KILL

Destroys the windows associated with the task. This is the same as selecting End Task from the Task Manager.

Syntax **KILL *taskid***

Parameters taskid The task to kill

Note This does NOT give the application a chance to save any unsaved work, and in some cases may make the system unstable.

Example

```
KILL After Dark
```

This will force After Dark to close.

See Also

CLOSE, NUKE

LH

Note This command has been superseded by the RUN -high command

Runs the specified file like the RUN command, but tries to keep it from using as much low memory.

Syntax **LH file [*arguments*]**

Note This works better for some programs than for others; for example, `MOD4WIN` still insists on stealing all low memory even when it's started with **LH**.

Example `LH BCW`

Loads Borland C++, and forces it to use as little low memory as possible.

See Also

RUN

LOAD

Note This command has been superseded by the RUN -min command

Loads the file, as if you selected File Run from the Program Manager with the Run Minimized box checked, or placed the file on the load= line in `win.ini`. The application will be started minimized, and the currently active application will stay active. This is a nice way to start Windows applications from a full-screen DOS prompt without switching away from the DOS prompt.

Syntax **LOAD *file* [*arguments*]**

Example

```
LOAD CLOCK
```

Loads the Clock program as an icon.

See Also

RUN

LOWER

Pushes a window to the bottom of the pile. The currently active application stays active.

Syntax **LOWER *windowid***

Parameters windowid The window to lower.

Environment Sets `%UCOUTPUT` to the *windowid* of the lowered window.

Example `LOWER Notepad`

Pushes Notepad beneath all other windows.

See Also

RAISE, UNSTICK

MAX

Maximizes and activates a window. For most applications, maximizing a window makes it fill the entire screen. DOS applications may not take up the whole screen when maximized, depending on their font size.

Syntax **MAX *windowid***

Parameters windowid The window to maximize.

Environment Sets `%UCOUTPUT` to the *windowid* of the maximized window.

Example

```
MAX Microsoft Word - UNIXCORN.DOC
```

Makes Microsoft Word take up the whole screen.

See Also

MIN, RESTORE

MEM

Prints information on Windows' use of memory.

Syntax **MEM**

Environment Sets `%UCOUTPUT` to the amount of free virtual memory, in KB.

Example

```
MEM
WINDOWS MEMORY INFORMATION:
WINDOWS MEMORY INFORMATION:
Physical memory:             11372 KB
Swapfile size:                4096 KB (temporary using file F:\
WIN386.SWP)
Total virtual memory:        15468 KB

Global heap items used:       1941 (931 discardable, 329 free, 6251
unused)

Windows code:                 4397 KB (336 KB low)
Windows data:                 3467 KB (156 KB low)
Total Windows usage:          7864 KB (493 KB low, 4906 KB discardable)

DOS and system usage:         1367 KB

Free virtual memory:         11451 KB
Free lockable memory:         9620 KB
Free before discard:          6236 KB
Free physical memory:         1520 KB
Free low memory:            628929 KB

Free resources:            GDI     49%
                           USER    62%
```

Physical memory:	The amount of RAM memory which Windows controls. This is the total memory installed on your computer, less the memory used by DOS, the Windows system, TSR programs, disk caches, and the like.
Swapfile size:	The size and location of Windows' swapfile on your hard disk. Unixcorn will attempt to determine if the swapfile is permanent or temporary.
Total virtual memory:	The total amount of memory Windows has to play with.
Global heap items:	The number of items (blocks of code and data) on Windows' memory heap. Discardable items will be thrown out to make more memory if necessary, and then reloaded from disk later as needed. You can force Windows to throw out all discardable items with the COMPACT command. If you run out of unused items, Windows will not be able to allocate any more memory.
Windows code:	The total amount of code used by all currently loaded Windows applications.
Windows data:	The total amount of data used by all currently loaded Windows applications.
Total Windows usage:	The total amount of memory being used by all Windows applications. Discardable memory will be cleared if more free space is needed by an application. You can force Windows to

clear all discardable memory with the COMPACT command. Does not include memory used by the Windows system itself (see below).

DOS and system usage: The total amount of memory being used by all DOS applications and the Windows system.

Free virtual memory: The maximum amount of memory Windows could make available. To get this much memory, Windows would have to clear all discardable items.

Free lockable memory: The maximum amount of physical memory (RAM) which Windows could make available. To get this much memory, Windows would have to swap everything else to the swapfile. Running several DOS applications in the background can really reduce this value, since Windows can't swap them out to the swapfile.

Free before discard: The maximum amount of memory Windows can allocate before it will have to start clearing discardable items.

Free physical memory: The amount of unused physical memory. When this hits zero, Windows will start swapping memory to the swapfile on your hard disk.

Free low memory: Windows has a limited amount of free DOS memory which can be allocated by programs. If this hits zero, you may be unable to start new applications even though you've got megabytes of free virtual memory and lots of system resources left. Microsoft Word and Excel seem to need a fair amount of this memory, as does Borland C++.

Free resources: Windows keeps track of resources like fonts, window handles, cursors, and icons in two places called the GDI and USER heaps. When these get low, programs may refuse to load or run, often suggesting that you may need to close other applications first. Some programs also have bugs (memory leaks) which gradually consume resources over time.

See Also

LH, HEAPWALK, COMPACT

MIN

Minimizes a window (turns it into an icon).

Syntax **MIN *windowid***

Parameters windowid The window to minimize.

Environment Sets `%UCOUTPUT` to the *windowid* of the minimized window.

Example `MIN *`

Lets you pick a window to minimize.

See Also

MAX, RESTORE

MODULES

Lists all currently loaded modules. A module is a piece of a program; some programs may use several modules to hold all their code.

Syntax **MODULES [*module extension*]**

Parameters ***module-extension*** You can specify a file extension (e.g. DLL or FON) to list only those modules with matching extensions.

Environment Sets `%UCOUTPUT` to the number of modules printed.

Note If a program crashes badly or is ended with the NUKE command, it may not get a chance to unlock all of the modules it was using. In this case, the lock counts for those modules may be too high. This can consume additional memory. You can release these modules using the UNLOAD command.

Example

```
MODULES
Module Name          Code     Data Lock  Filename
   279 KERNEL       60992    33472   51  C:\WIN3\SYSTEM\KRNL386.EXE
   615 COMM          6528     1600   38  C:\WIN3\SYSTEM\COMM.DRV
   903 FONTS            0     6144    2  C:\WIN3\SYSTEM\VGASYS.FON
   967 GDI         203488  1726336   38  C:\WIN3\SYSTEM\GDI.EXE
   983 USER        234304    85536   37  C:\WIN3\SYSTEM\USER.EXE
  2015 DISPLAY     221440   208448    1  C:\WIN3\SYSTEM\MACH.DRV
  6135 WIFEMAN      31200    22368    1  C:\WIN3\SYSTEM\WIFEMAN.DLL
  1447 BACKMENU     30720    30176    1  C:\W\SYS\BACKMENU.EXE
  5607 BACKMLIB      7424    12160    1  c:\w\sys\BACKMLIB.DLL
  5567 COMMDLG       1248     5632    7  C:\WIN3\SYSTEM\COMMDLG.DLL
  5599 PMGROUPS      5216     9824    1  c:\w\sys\PMGROUPS.DLL
```

(output truncated to save space in the manual)

Module is the handle of the module you would use to see a single module's memory using the HEAPWALK command. Name is the module name. If more than one task uses the same module (for example, two DOS Prompts), the same name may occur more than once in the list. The next two fields are the number of bytes of code and data the module uses. The Lock count is how many functions from the module are currently being used. The last field is the file the module was loaded from.

Example

```
MODULES EXE
Module Name          Code     Data Lock  Filename
   279 KERNEL       60992    33728   51  C:\WIN3\SYSTEM\KRNL386.EXE
   967 GDI         203488  1734400   38  C:\WIN3\SYSTEM\GDI.EXE
   983 USER        234304    85984   37  C:\WIN3\SYSTEM\USER.EXE
  1447 BACKMENU     30720    30176    1  C:\W\SYS\BACKMENU.EXE
  2287 DROPPER      19008    52032    1  C:\W\SYS\DROPPER.EXE
  2279 MSGICON      10080    12672    1  C:\W\SYS\MSGICON.EXE
  4983 AD          101568    40000    1  C:\WIN3\AFTERDRK\AD.EXE
 13279 WINWORD    1422464    75872    1  C:\APPS\WFW\WINWORD.EXE
 22047 UNIXCORN     19808    22240    1  C:\BCC\W\UNIXCORN.EXE
```

Lists only modules with a file extension of .EXE.

See Also

MEM, HEAPWALK, UNLOAD

NUKE

Forces the task to exit abruptly, similarly to pressing Ctrl+Alt+Del.

Syntax **NUKE [-f] *taskid***

Parameters taskid The task to nuke

Switches **-f** Forces Unixcorn to nuke the task, without asking for confirmation.

Note Nuking a task is dangerous enough that Unixcorn will ask you if you are sure you want to do this. However, sometimes a task can't be KILLed and this is your last resort. Tasks that are nuked also may not have a chance to free memory or system resources they were using. **Any time you nuke a task, you should save your work in all other applications and restart Windows.**

Example `NUKE Microsoft Excel`

Terminates Microsoft Excel with extreme prejudice.

See Also

CLOSE, KILL

PASTE

Outputs the current clipboard text. The output is limited to about 30 KB (around 400 full lines of text).

Syntax **PASTE**

Note This command only works with UC.

Example `UC PASTE > CLIPBRD.TXT`

This will copy the text on the clipboard to the file `CLIPBRD.TXT`.

See Also

CLEAR, COPY, CUT

PAUSE

Changes whether output to the DOS prompt pauses every screen full. By defaultIf you turn on pausing,

Syntax **PAUSE [on | off]**

Parameters

on	Turn pausing on. After each screen full, UC will ask you to hit a key to display the next screen. After each screen full, you may also press the Escape key to stop printing output.
off	Turn pausing off. Unixcorn will dump all its output in one big long stream. Use this setting if you want to pipe the output of UC to a file or another program.

If neither ***on*** nor ***off*** is specified, toggles pausing (from ***off*** to ***on***, or ***on*** to ***off***).

Environment Sets `%UCOUTPUT` to 1 if pausing is on or 0 if pausing is off.

Example

```
PAUSE ON
```

Turns pausing on.

See Also

Using Unixcorn from DOS

PLAY

Plays the specified wavefile. The file should be in RIFF WAVE format (if it ends in `.WAV`, it probably is).

Syntax **PLAY [*wavefile*]**

Parameters

wavefile The wavefile to be played. This may be a filename (ex: `ohoh.wav`) or one of the Windows system sounds (`systemasterisk`, `systemhand`, `systemdefault`, `systemexclamation`, `systemquestion`, `systemexit`, `systemstart`). If no wavefile is specified, plays the default beep noise set in the Control Panel (`systemdefault`).

Example

```
PLAY OHOH.WAV
```

Plays `OHOH.WAV`.

Example

```
PLAY SYSTEMEXIT
```

Plays the sound Windows makes when it exits.

See Also

ECHO

POS

Repositions a window. Hidden windows may not be repositioned.

Syntax **POS *x,y windowid***

Parameters

x	X-coordinate of the window, measured from the left side of the screen if ***x*** is positive or the right hand side if ***x*** is negative.
y	Y-coordinate of the window, measured from the top of the screen if ***y*** is positive or the bottom if ***y*** is negative.
windowid	The window to reposition.

Environment Sets `%UCOUTPUT` to the *windowid* of the repositioned window.

Example

```
POS 0,0 Microsoft Word - UNIXCORN.DOC
```

Moves Microsoft Word to the upper left hand corner of the screen.

Example

```
POS -1,-1 Microsoft Word - UNIXCORN.DOC
```

Moves Microsoft Word to the lower right hand corner of the screen.

See Also

SIZE

PROMPT

Starts UcWin with the Unixcorn Command Line dialog.

Syntax **PROMPT**

Note This command only works with UcWin. This command can't be used in a batch file.

Example A Program Manager item with the command line "`UCWIN PROMPT`"

See Also

Using Unixcorn from Windows, UcWin

QUIT

Quits Unixcorn. Both the DOS and Windows parts of Unixcorn will exit.

Syntax **QUIT**

Example `QUIT`

See Also

EXIT, REBOOT, RESTART

RAISE

Temporarily pulls the window to the top of the desktop. The currently active application stays active. This lets you keep working in your current application, but see the contents of another window that might otherwise be buried. Use the STICK command to keep a window on top all the time.

Syntax **RAISE *windowid***

Parameters windowid The window to raise.

Environment Sets `%UCOUTPUT` to the *windowid* of the raised window.

Example

```
RAISE Notepad
```

Pulls Notepad above MS-DOS Prompt; MS-DOS prompt stays active.

See Also

ACTIVATE, LOWER, RESTORE, STICK

REBOOT

Exits Windows and reboots the computer, as if you pressed Ctrl+Alt+Del twice.

Syntax **REBOOT**

Note You will be asked to confirm this command before Unixcorn actually reboots the computer.

Example `REBOOT`

See Also

EXIT, QUIT, RESTART

RESTART

Exits and restarts Windows. This is a good thing to do after you've used NUKE on an uncooperative application.

Syntax **RESTART**

Note You will be asked to confirm this command before Unixcorn actually restarts Windows.

Example `RESTART`

See Also

EXIT, QUIT, REBOOT

RESTORE

Activates a window in its normal position, neither minimized nor maximized.

Syntax **RESTORE *windowid***

Parameters windowid The window to restore.

Environment Sets `%UCOUTPUT` to the *windowid* of the restored window.

Example

```
RESTORE Microsoft Word
```

Restores Microsoft Word from an icon to normal size

See Also

MAX, MIN

RUN

Runs the specified file, as if you selected File Run from the Program Manager, or placed the file on the run= line in `win.ini`. If the file is a data file that has a program associated with it in File Manager, you can run it directly.

Syntax **RUN [-min | -max | -hide] [-high] [-wait] [-in *directory*] *file* [*arguments*]**

Parameters	***file***	The filename of the application or document to run
	arguments	Optional arguments to the program
Switches	**-min**	Loads the file, as if you selected File Run from the Program Manager with the Run Minimized box checked, or placed the file on the load= line in `win.ini`. The application will be started minimized, and the currently active application will stay active. This is a nice way to start Windows applications from a full-screen DOS prompt without switching away from the DOS prompt.
	-max	Runs the application maximized (with its window covering the whole screen).
	-hide	Runs the application hidden. You can get a list of hidden windows with the WINDOWS -h command, and can unhide a window with the SHOW command.
	-high	Runs the specified file, but tries to keep it from using as much low memory. This is more effective with some programs than with others.
	-wait	Waits until the program exits before returning. This is useful in batch files where you want to run a series of programs sequentially (for example, you might run one program which creates a datafile and then another to print the datafile). Using UnixcTSR will improve the performance of this option.
	-in *directory*	Starts the application in the specified directory.

Environment Sets `%UCOUTPUT` to the *taskid* of the program which was started.

Example
```
RUN CONTROL.EXE
```

Starts the Control Panel

Example
```
RUN WRITE README.WRI
```
Example
```
RUN README.WRI
```

Either of these starts Write with the file `README.WRI` loaded.

Example
```
RUN -hide NOTEPAD /P FOO.TXT
```

Prints `FOO.TXT` using Notepad, without showing Notepad's window on the screen.

Example
```
RUN -in C:\DOCS WRITE
```

Starts Write so that the directory you first see when you select File Open from Write's menu is `C:\DOCS`.

Example

```
RUN -wait WRITE FOO.TXT
ECHO Done editing FOO.TXT!!!
```

Edits `FOO.TXT`, waiting until Write exits before popping up the message box.

See Also

LOAD, LH, RUNHIDDEN

RUNHIDDEN

Note This command has been superseded by the RUN -hide command

Runs the specified file like the RUN command, but hides the program's main window. You can get a list of hidden windows with the WINDOWS -h command, and can unhide a window with the SHOW command.

Syntax **RUNHIDDEN file [*arguments*]**

Example

```
RUNHIDDEN NOTEPAD /P FOO.TXT
```

Prints `FOO.TXT` using Notepad, without showing Notepad's window on the screen.

See Also

RUN

SHOW

Shows a previously hidden window.

Syntax **SHOW *windowid***

Parameters windowid The window to show.

Environment Sets `%UCOUTPUT` to the *windowid* of the shown window.

Example `SHOW Unixcorn`

Re-shows Unixcorn's icon.

See Also

HIDE, RUNHIDDEN

SIZE

Resizes a window. Minimized and maximized windows may not be resized (they're already as small or big as they can get). The size includes the space used by the title bar, scrollbars, etc.

Syntax **SIZE *width,height windowid***

Parameters

width	The new window width in pixels.
height	The new window height in pixels.
windowid	The window to resize.

Environment Sets `%UCOUTPUT` to the *windowid* of the resized window.

Example

```
SIZE 500,300 Microsoft Word
```

Sizes Microsoft Word to 500 pixels wide by 300 pixels high.

See Also

POS

STICK

Sticks the window to the monitor glass. The window will stay on top of all other windows, regardless of which application is active. This is what the "Always On Top" option of Clock does.

Syntax **STICK *windowid***

Parameters windowid The window to stick.

Environment Sets `%UCOUTPUT` to the *windowid* of the stuck window.

Example `STICK Unixcorn`

Keeps the Unixcorn icon visible, even if another application is maximized.

See Also

RAISE, UNSTICK

TASKS

Lists the tasks currently running under Windows.

Syntax **TASKS [-w] [-a] [-f]**

Switches

-w Lists all named windows for each task.

-a Lists all named and unnamed windows for the task.

-f Lists the file each of the tasks was run from.

Environment Sets `%UCOUTPUT` to the number of tasks printed.

Example

```
TASKS
 Task Parent  Module   Window  Title
 1431    319  BACKMENU   4796  Back-Menu
 2303   1431  DROPPER    5864  Dropper
 7431   1431  AD        15488  After Dark
 6735   1431  BARCLOCK  65535  (no name)
13711   2303  WINWORD   20196  Microsoft Word - UNIXCORN.DOC
14063   2303  WINOLDAP  18552  MS-DOS Prompt
19463   2303  UNIXCORN  24268  Unixcorn
```

The Task field gives the task number for the task. This is the number you would use as the *taskid* parameter for commands like CLOSE and KILL. The task number for a program may be different if you exit the program and restart it (for example, exiting and restarting Microsoft Word changed its task number to `14423`).

Parent is the task which started this task (in the example, BACKMENU started DROPPER). The module is where Windows stores the code and data for the task.

Given no options, Unixcorn will attempt to guess the *windowid* and title of the task's main window. You can get a list of the windows associated with a task using the -w or -a options (see below).

If -w is specified, lists all named windows for the task. For example, the WINWORD task now lists the following:

```
13711   2303  WINWORD
                      20796   Dde Common
                      20196   Microsoft Word - UNIXCORN.DOC
```

If -a is specified, lists all named and unnamed windows for the task.

```
13711   2303  WINWORD
                      22516   (no name)
                      22860   (no name)
                      23204   (no name)
                      22104   (no name)
                      24080   (no name)
                      20196   Microsoft Word - UNIXCORN.DOC
                      21536   (no name)
                      20300   (no name)
                      20932   (no name)
                      20796   Dde Common
```

If -f is specified, lists the file each of the tasks was run from.

```
 Task Parent  Module    File
 1431    319  BACKMENU  C:\W\SYS\BACKMENU.EXE
 2303   1431  DROPPER   C:\W\SYS\DROPPER.EXE
 6735   1431  BARCLOCK  C:\W\SYS\BARCLOCK.EXE
 7431   1431  AD        C:\WIN3\AFTERDRK\AD.EXE
13711   2303  WINWORD   C:\APPS\WFW\WINWORD.EXE
14063   2303  WINOLDAP  C:\WIN3\SYSTEM\WINOA386.MOD
19463   2303  UNIXCORN  C:\BCC\W\UNIXCORN.EXE
```

Note that for DOS tasks, the filename given is not the PIF file. Anyone know a fix for this?

See Also

CLOSE, KILL, NUKE

TITLE

Changes the title of a window.

Syntax **TITLE '*newtitle*' *windowid***

Environment Sets `%UCOUTPUT` to the *windowid* of the retitled window.

Parameters **'*newtitle*'** The new window title. Must be enclosed in single quotes.

windowid The window to restore.

Example

```
TITLE 'My Favorite Program' Unixcorn
```

Renames the Unixcorn icon to "My Favorite Program"

UNLOAD

Unloads a module from memory. This can be useful if a program crashes badly or is terminated abruptly (such as when ended by the NUKE command) and doesn't free up all its resources. **Use this command with care - it can make the system unstable!**

Note You will be asked to confirm this command before Unixcorn actually unloads the module.

Syntax **UNLOAD [-f] *moduleid***

Parameters moduleid The module to unload.

Switches **-f** Forces Unixcorn to unload the module, without asking for confirmation.

Example

```
UNLOAD unixcorn
```

Unloads the module containing the Unixcorn code. Note that this will cause Unixcorn to crash.

See Also

MODULES, NUKE

UNSTICK

Reverses the effects of STICK. Other windows may now overlap the unstuck window.

Syntax **UNSTICK *windowid***

Parameters windowid The window to unstick.

Environment Sets `%UCOUTPUT` to the *windowid* of the unstuck window.

Example `UNSTICK Unixcorn`

Allows other windows to be on top of the Unixcorn icon.

See Also

LOWER, STICK

UPTIME

Prints how long Windows has been running.

Syntax **UPTIME**

Environment Sets `%UCOUTPUT` to the number of seconds Windows has been running.

Example

```
UPTIME
Breakdown of time spent since Windows started:

What                                  Time
---------------------------------------
Windows applications:              8:54:41
DOS applications:                  4:48:06
---------------------------------------
Total Windows uptime              13:42:48
```

The example shows a system that has been up almost two hours. Of that time, almost nine hours has been spent in Windows applications like Microsoft Excel, Minesweeper, and After Dark. The remaining five hours has been spent running full-screen DOS applications.

USERMENU

Sets or shows Unixcorn user menu items.

Syntax **USERMENU [*num*] ['text' *command*]**

Parameters

num	The menu slot number to create or delete. Must be between 0 and 9. If not specified, lists the command assigned to each menu slot. If the specified slot is already used, the new menu item will overwrite the old one.
'text'	The text that will appear in the menu item. This must be enclosed in single quotes. If not specified, the menu slot is deleted. If the text contains an ampersand "&", the character after the ampersand will be underlined.
command	The command to be executed when the menu item is selected.

Example

```
USERMENU
USERMENU items:
1 'Notepad' run notepad.exe
4 'Task list' tasks
5 'Memory info' mem
```

Lists all used menu slots.

Example

```
USERMENU 2 'Stick &Window' STICK *
```

Creates a user menu item in slot 2 with the text "Stick Window". When the menu item is selected, you can pick a window to stick to the monitor glass. You could easily define another user menu item to "Unstick window"; this would allow you to stick and unstick windows using only the mouse.

Example

```
USERMENU 2
```

Deletes the user menu item in slot 2.

See Also

Creating Aliases and Menu Items for Common Commands
ALIAS

WALLPAPER

Sets the background wallpaper.

Syntax **WALLPAPER [-c | -t | -n] [*bitmapfile*]**

Switches

-c Place the bitmap in the center of the screen.

-t Tile the bitmap on the screen.

-n No wallpaper - removes the background wallpaper.

Example `WALLPAPER -C 256color.bmp`

Centers the bitmap `256color.bmp` on the Windows background. This mode works best for bitmaps that are around the same size as your screen.

Example `WALLPAPER -T marble.bmp`

Tiles the bitmap `marble.bmp` on the Windows background. This mode works best for small repetitive bitmaps.

Example `WALLPAPER -N`

Removes the background wallpaper. The Windows background is now blank.

WHAT

Prints a single piece of information about Windows.

Syntax **WHAT [screen | ver | winver | gdires | userres | res | mem | physmem]**

Switches

screen	Screen resolution (example: 1024x768x256)
ver	Unixcorn version (example: 2.0)
winver	Windows version (example: 3.11)
gdires	Percentage of GDI resources free
userres	Percentage of USER resources free
res	Percentage of system resources free
mem	Free virtual memory, in KB
physmem	Free physical memory, in KB

Environment Sets `%UCOUTPUT` to the piece of information.

Example

```
WHAT SCREEN
1024x768x256
```

Prints the current resolution Windows is running in.

See Also

UPTIME, MEM

WINDOWS

Lists all top-level windows.

Syntax **WINDOWS [-h] [-c] [-p] [-m]**

Switches

-h Also list hidden windows.

-p Print each window's position. This is the default if neither **-c** nor **-m** is specified.

-c Print each window's class.

-m Print the module and program which owns each window.

Environment Sets `%UCOUTPUT` to the number of windows printed.

Example

```
WINDOWS
Window   Displayed  Position    Size    Title
------   ---------  --------    ----    -----
 17536  +minimized   246,696    36x 36  Rainbow Pad - C:\DOCS\TTD.PAD
 16944   Normal      136,229   630x386  UcWin - win
  7292   minimized    96,696    36x 36  Unixcorn
 20984   Normal       44, 40   858x535  Microsoft Word - UNIXCMDS.RTF
 13484   minimized   171,696    36x 36  4DOS Prompt

Active window: 16944  UcWin - win
```

The Window field gives the handle for the window. This handle will most likely change if you close and restart the application; for most of the commands in the following section it's probably better to use the window title.

Displayed is either Normal, minimized, MAXIMIZED, or (hidden), depending on how the window appears on-screen. A plus "+" before the displayed column indicates the window has been STUCK to the monitor glass or is running with its own "Always on Top" option enabled. The position and size of the window are given if applicable. All minimized windows have the same size because they are icons. Finally, the window's title is listed.

If **-h** is specified, all hidden windows are also listed. Hidden windows do not have positions or sizes.

```
Window   Displayed  Position    Size    Title
------   ---------  --------    ----    -----
 17536  +minimized   246,696    36x 36  Rainbow Pad - C:\DOCS\TTD.PAD
 16944   Normal      136,229   630x386  UcWin - win -h
  7292   minimized    96,696    36x 36  Unixcorn
 21312   (hidden)       n/a      n/a    Dde Common
  6832   (hidden)       n/a      n/a    ATIKey Hook
  6212   (hidden)       n/a      n/a    After Dark
  6296   (hidden)       n/a      n/a    PASSWORD
 13484   minimized   171,696    36x 36  4DOS Prompt
 20984   minimized   321,696    36x 36  Microsoft Word - UNIXCMDS.RTF

Active window: 16944  UcWin - win -h
```

If **-c** is specified, the class of each window is listed. If you want to see each window's position at the same time, use **-c -p**. All DOS prompts have class `tty`. This

information probably isn't very useful unless you're a programmer.

```
Window  Class            Title
------  -----            -----
 17536  RbPad            Rainbow Pad - C:\DOCS\TTD.PAD
 16944  UcWin            UcWin - win -c
  7292  Unixcorn         Unixcorn
 13484  tty              4DOS Prompt
 20984  OpusApp          Microsoft Word - UNIXCMDS.RTF

Active window: 16944  UcWin - win -c
```

If **-m** is specified, the module and program controlling each window is listed. If you want to see each window's position at the same time, use **-m -p**. All DOS prompts are controlled by `WINOLDAP`. This information probably isn't very useful unless you're a programmer.

```
Window  Modid Module   Filename                 Title
------  ----- ------   --------                 -----
 17536  10887 RBPAD    D:\BCC\RBPAD\RBPAD.EXE    Rainbow Pad - C:\DOCS\
TTD.PAD
 16944  11255 UCWIN    D:\BCC\W\UCWIN\UCWIN.EXE  UcWin - win -m
  7292  10039 UNIXCORN D:\BCC\W\UNIXCORN.EXE     Unixcorn
 13484  10271 WINOLDAP C:\WIN3\SYSTEM\WINOA386.  4DOS Prompt
 20984   2439 WINWORD  C:\APPS\WFW\WINWORD.EXE   Microsoft Word -
UNIXCMDS.RTF

Active window: 16944  UcWin - win -m
```

See Also

CLASSES, MODULES, TASKS

Popups

Where a *taskid* is required, you may enter either the task number given by the TASKS command (ex: `13711`), the handle of a window (ex: `20196`) or its title (ex: `Microsoft Word - UNIXCORN.DOC`). You do not need to type the whole title, just enough so that Unixcorn knows which window you mean. It doesn't matter if you don't capitalize the window name. For example, `MICROSOFT WORD` would select the same window. Be careful about typing enough of the title - just typing `M` would match `Microsoft Word`, `MS-DOS Prompt`, or any other window beginning with the letter `M`.

You may enter a single plus (+) for *taskid*, which will match the currently active task.

You may also enter a single asterisk (`*`) for *taskid* or not specify anything at all. Unixcorn will change the cursor to a Unicorn horn, and will pop up a message box with the currently selected window's title and icon. You can then pick a window by clicking the left mouse button on it. You can abort the command by clicking the right mouse button anywhere.

The first 1024 KB of memory. If no low memory is available, you will not be able to start any more programs.

The shell is the task responsible for starting and exiting Windows. It usually has the lowest task number.

Where a *windowid* is required, you may enter either the handle of a window (ex: `20196`) or its title (ex: `Microsoft Word - UNIXCORN.DOC`). You do not need to type the whole title, just enough so that Unixcorn knows which window you mean. It doesn't matter if you don't capitalize the window name. For example, `MICROSOFT WORD` would select the same window. Be careful about typing enough of the title - just typing `M` would match `Microsoft Word`, `MS-DOS Prompt`, or any other window beginning with the letter `M`.

You may enter a single plus (+) for *windowid*, which will match the currently active window.

You may also enter a single asterisk (`*`) for *windowid* or not specify anything at all. Unixcorn will change the cursor to a Unicorn horn, and will pop up a message box with the currently selected window's title and icon. You can then pick a window by clicking the left mouse button on it. You can abort the command by clicking the right mouse button anywhere.

A *moduleid* may be specified as the handle of a module as listed in the MODULES command (i.e. `279`) or its name (i.e. `KERNEL`).