# Clipper Documentation

**COLLABORATORS**

| | | | |
|---|---|---|---|
| | *TITLE* : Clipper Documentation | | |
| *ACTION* | *NAME* | *DATE* | *SIGNATURE* |
| WRITTEN BY | | February 12, 2023 | |

**REVISION HISTORY**

| NUMBER | DATE | DESCRIPTION | NAME |
|---|---|---|---|
| | | | |

# Contents

# Chapter 1

# Clipper Documentation

## 1.1 All the Clipper buttons

Clipper v1.01

## 1.2 New features

New features in v1.1

This version of Clipper has been a rush and many things haven't made it in yet or weren't completely finished. A new version will appear around Easter time. See the to do section for more details.

Here's a list of what did make it in...

Big things:

1) Programs using SendIO can now be intercepted along with the vast majority of programs using DoIO.

2) Clip view windows notice when data changes i.e. if you have a clip view window open and you change things in another window or another application pastes to the clipboard the window will update clipboard contents accordingly.

Little things:

1) The escape key now can be used to cancel changes in the prefs window.

2) Also, the escape key now denies rather than passes through clip requests (which probably is more in tune with your wishes when you press the escape key.)

3) For similar reasons, the close gadget also now denies rather than passes through clip requests.

4) Hot keys added for gadgets that were missing them in the prefs window.

## 1.3 Introduction to Clipper

Introduction

Clipper's lot in life is to add proper multi-clipboard support to the Amiga's operating system. Generally application clipboard implementation is poor and short sighted, but it can be better; and Clipper does three big things to improve matters.

1) A graphical display of the contents of as many clip units as you wish. This is done using the datatypes system of v39+. Thus graphics show up as graphics, text as text etc, etc. Anything you have a datatype for will be shown.

2) An easy clipboard managing system. Using the above graphical representation you can view, delete, move, copy or save to a file the contents of any clip unit.

3) Lastly and best of all, Clipper allows all your current programs to have dynamic multi-clipboard support. Clipper will intercept clipboard reads and writes, bring up a window showing all the clip units, and ask you which clip unit you wish to read from or write to.

Go to the features page for a quick list of Clipper's good points.

## 1.4 Clipper's good points

Quick List Of Clipper Features

* Graphical representation of the contents of as many clipboard units as you want to see. Provides windows like this.

* Clip view window summonable at any time by hot key.

* Interception of clipboard reads and writes by *ALL* applications, allowing re-direction of a read/write request to a unit of your choice picked from that nice clip view window.

* Viewing, saving, deleting, copying and moving of clip units.

* Uses the v39+ datatypes system, allowing representation of any type of data you have a datatype for: 8SVX sounds, amigaguide files, text documents, ILBM,JPEG,GIF,24bitILBM, fonts, executable files etc,etc.

* Configurable via built in GadTools (not Mui ;-) ) preferences editor. Change size of clip unit representations, intercept qualifier key, screen for the full clip view mode etc,etc.

* A commodity.

* Completely proper and re-entrant code allowing for many different clip requests and clip view windows simultaneously.

* Proper and safe semaphore protection to ensure all requests fully complete before (the unlikely event of you) quitting Clipper.

* Mentioning of Haille Selassie.

## 1.5 What you need...

Clipper's Requirements

Clipper absolutely must have the following...

V39 or better of Amiga OS fully installed.

68020 or better.

Very highly recommended is a system featuring...

Your Clips: assign either being to RAM: or a hard disk to keep the speed up.

A faster than standard Amiga (i.e. 14mhz 020 with only chip RAM) to make the OS'S datatypes system fast enough.

And probably of use...

A commodity exchange program for when you forget the spurious hot key to summon the clip prefs window.

## 1.6 How to install Clipper

Installation

By double clicking on the "InstallClipper" icon you can use the supplied installer script to place Clipper on your hard drive. Those users who are running from a floppy based system or who have a grudge against the installer can easily install Clipper manually: Just copy the "Clipper" and "Clipper.info" files to either "SYS:WBStartup" if you want Clipper run whenever the Amiga boots (recommended), or somewhere such as "SYS:Commodities" if you are only going to use Clipper occasionally and don't want it run automatically. Note that if you are installing this guide file somewhere by hand you should copy the guide file itself and the "ClipViewWindow.ilbm" file to the same directory.

Once that's done Clipper should be ready to go. Clipper will report most possible envisaged problems to you, but check the requirements section for a formal list of what you need.

## 1.7 Using Clipper

Using Clipper

Main functionality:

Once Clipper has been successfully installed there are two ways in which you will use it. Firstly, by pressing a hot key (ctrl lalt c by default) you can summon up a window containing a view of all the clip units. Secondly you can use Clipper to intercept and re-direct clipboard reads and writes from all applications. In both cases you will be presented with a window showing a number of clipboard units. The windows function virtually identically, although functions concerned with re-directing clip requests are obviously restricted to windows that actually represent a request as opposed to having been summoned by hot-key. This main clipboard window is described here .

Preference settings:

The way Clipper behaves including the number of clip units to show in a window, the size of each clip unit in the window, default window size, and when to intercept clip requests can be changed with Clipper's preferences window .

Clipper's preferences are saved as tool types in the Clipper.info file and you thus may edit them by selecting the Clipper icon on your Workbench and selecting the "information" option from the "icon" menu of the Workbench. Quite why you should want or need to do this is not clear, and some of the preferences will seem slightly cryptic this way. Better to use that prefs window.

## 1.8 Clipper's main clip view windows

Clipper's Clip View Windows

What you'll see

A clip view window will look something like this.

When you'll meet a clip view window

You will encounter one of Clipper's clip view windows when you summon one up by hot key (ctrl lalt c by default) or Clipper intercepts a clipboard read or write. The screen the window will appear upon will depend on your preference settings - by default clip windows are opened on the frontmost screen if that is a public screen and if it is not a public screen then on the default public screen. This is the recommended way to use Clipper, although windows can be forced to appear on the front screen whether it is public or not (danger!) or always on the default public screen.

A description of a clip view window

In a clip view window you will be shown the contents of a number of clipboard units. The number of clip units, and the size of their representation is changeable via the preferences editor. Any data that you have a datatype for will be correctly represented. Everyone should have the ilbm, text, 8svx(sound) etc datatypes supplied with v39 of the OS, but many more datatypes are available in the public domain (JPEG,GIF,24bitILBM,Executable,info file,font etc,etc). Typically the whole of the data in the clipboard will not fit in the space provided and many data types allow you to scroll the data around by clicking the left button on the image of the data and then, while holding the button down, moving the mouse outside the datatype object in the direction you wish to scroll. Other datatypes behave in different ways - for example the 8svx sound datatype plays the sample it represents when you click on it.

If you have enabled the gadget help option a message box will appear in the window providing a little information about the functionality of each gadget when you hold the mouse over it. Additionally, you will be told the type of data in each clip unit whenever the mouse is over a unit.

You may resize the window at any time, and Clipper will always display as many units as can be fitted into the window up to the user defined maximum level that is initially set in the preferences window, but also may be changed for each window by typing a new number into the "clips" integer gadget in each clip view window (click on the gadget, or press l to select it.)

Initially all the clip units in the window will have "wait" written in them, changing to a view of the clip unit contents or "empty" as each unit is built up in turn. This initial refresh, and indeed all other refreshes, are fully interruptible. Clipper allows you to do all the normal things you can do in a clip view window even when the window is being constructed. This makes Clipper a lot quicker, as you can direct clipboard reads/writes etc (or indeed do whatever else it is you want to do) without waiting for the window to fully refresh (which can take a while if there's big graphic images etc floating about.) Do note however that Clipper may sometimes not respond to your input immediately when a window is being refreshed. This is because Clipper cannot actually check for input while a datatype object is being made, although it can check while an object is being laid out and refreshed which actually takes up by far the greatest amount of refresh time. Clipper will "notice" all input and process it just as soon as it can.

Clip view windows also notice when clipboard contents are changed either by another task or your input to another clip view window. The clip view windows will update themselves when the data in the clipboard changes, and again you will lose instant response to your input during this time.

Selecting and operating on a clip unit

Clipper has a notion of a currently selected clip unit. You may select a unit by clicking on it with the mouse, or (for units 0 to 9) by holding down the right Amiga key and pressing the appropriate number. You can change the selected unit by clicking on another unit, selecting a new unit using the right Amiga and number keys, or by using the cursor keys. Once you have a unit selected, Clipper allows you to do various things with it...

1) You may view the clip unit in full screen mode on a screen specified by the user in the preferences window. Press v, right-Amiga-v, select view from the "Clip Unit" menu or click on the small "v" button in the clip view window to do this. Click on the close gadget on the window the clip is displayed in on the full screen to close the full clip view and return to Clipper's clip view window.

2) You may save the contents of the clip unit to a file by pressing s, right-Amiga-s, selecting "save" from the "Clip Unit" menu, or clicking the small s button. You will be presented with a file requester asking you where you want to save the file to.

3) You may delete the clip unit be selecting delete from keyboard, gadget or menu.

4) You may copy the unit to another unit by selecting copy from keyboard, gadget or menu. You will then need to select a second clip unit to copy to. Do this by clicking on a new clip unit or selecting it with the cursor and/or right-Amiga-number keys and then pressing return or enter.

5) You may move the clip unit to another unit by selecting move by keyboard, gadget or menu. Select a destination unit as for the copy operation above.

Copy and move operations may be aborted by pressing the escape key, or by pressing the small "A" gadget or selecting the "abort copy/move" option on the "clip unit" menu. The button and menu item will normally be ghosted out, and will only become available when a copy or move is in operation.

Closing the clip view window

You can close the clip view window at any time by clicking the close gadget, or pressing the escape key. Things are slightly more complex for windows not summoned by hot-key and representing a clip request - see below.

Clip view windows as intercepted clip requests

When a clip view window is opened as a result of intercepting a clipboard read or write the window retains all the functionality outlined above, but also has several other buttons and surprises.

You may select the unit you wish the application to actually perform the read from/write to by double clicking on a unit with the mouse, pressing the appropriate number key (for units 0 to 9), or by pressing return when a unit is selected (see above.) Pressing "n" allows you to actually type in the number of the clip unit you want to read from or write to. (Useful, for example, for clip units that aren't displayed - you can direct a request without having to re-size the window and double-click.) The gadget you type the number into after pressing the "n" key is unusually in replace mode (i.e. new characters overtype existing ones). This just speeds things up a little. However you go about it, this re-direction facility is Clipper's nicest feature and the reason that the project seemed like such a good idea.

You may also direct the completion of the clipboard read/write request in two other ways. Firstly, you may "pass through" the request (select "pass through" from the "clip request" menu, or press p or right-Amiga-p). Doing this simply tells Clipper to get out of the way and let the application perform the request as it originally intended. Secondly, you may deny the request ("deny" on the "clip request" menu, click the window close gadget, or press Esc, k or right-Amiga-k). This means that Clipper will tell the application that it cannot have access to any data in the clipboard and (so long as the application is properly written) it will report back that its clipboard read or write has failed.

Other miscellaneous bits about the clip view window

You may find out all you want to know about the clip request a clip view window represents by selecting the "request info" option from the "clip request" menu or by pressing i or right-Amiga-i. You will be told which application originally made the request, whether it is a read or write request and which clip unit was originally being read from or written to. Obviously, for hot-key summoned windows this information isn't available, and Clipper will tell you this in the information requester it brings up.

You may find out about Clipper by selecting the "About Clipper" menu option from the "Clip Request" menu, or by pressing right-Amiga-a. The requester is likely to include such interesting items as version number, author info, copyright notice and the number of years that a certain Ethiopian emperor has been dead for.

For a reminder of all the hot keys you can use, just press the HELP key in any clip view window.

And last and certainly least, the most obvious feature of all is the message window just below the number of clips gadget. You will be surprised and considerably enlightened to hear that this is where Clipper presents messages to you.

## 1.9 The preferences window

The Preferences Window

Summoning the window

You can summon up Clipper's preferences window using the Commodore commodity exchange program supplied with OS v39+ or any replacement commodity manager. Within your commodity manager program simply select the "show interface" option for Clipper. With the Commodore program you would do this by selecting Clipper in the list of commodities and then pressing the "show interface" button. Other commodity managers will operate similarly. Alternatively you may use the hot key to show/hide the preferences window - it is "ctrl lalt l" by default, but you may change this.

Using the window

Most of the buttons in the prefs window really should be self-explanatory, but here's the full run down of what they all do...

The "Clip Boxes Widths/Heights" buttons are the width and height in pixels of each clip unit that appears in a clip view window.

The "Window width/height" buttons define the default pixel width and height of each clip view window that appears, although these windows are re-sizeable at any time.

The "Max Clips Shown" button defines the maximum number of clip units you want shown in a clip view window. This number must be at least two and not above 256 as the OS supports 256 clip units. Illegal values will be truncated.

The "CX Priority" gadget allows you to defines Clipper's commodities priority relative to the other commodities you use. Commodities with higher priorities get to see input events including key presses earlier. In Clipper's case it should not be important to have a particularly high priority setting. As long as you take care not to "double-define" one of Clipper's hot keys identically to a key used by another commodity or application you should have no problems. If you do double-define a key, the commodity with the higher priority will get to hear about the key press while the commodity with the lower priority won't. Most users should be able to leave this setting well alone.

You may define the hot keys to pop up the Clipper preferences/control window and a clip view window with the "Clip view key" and "Show GUI key" respectively. Valid hot-key strings must contain at least one alpha-numeric character (roughly, all letter, number and punctuation keys) and are very likely to include a qualifier key as well. Valid qualifiers are as follows,

Lalt/Ralt = The left and right alt keys

LShift/RShift = The left and right shift keys

LAmiga/RAmiga = The left and right Amiga keys

Ctrl = The control keys

Clipper's checking of qualifier key names is purposefully not case sensitive, so that "ctrl" "CTRL" or even "cTrL" are all equivalent and valid. The case of letter keys is also not checked, so, for example, "LAlt l" is the the same as "LAlt L." If you want only a capital letter to act as a hot key simply include a LShift or RShift qualifier as a part of your hot key. Clipper will post a requester telling you if you have entered an invalid hot-key string.

The "Qualifier" cycle gadget specifies the qualifier key that is required in order to force Clipper to intercept a clipboard read or write. Selecting "none" here will cause Clipper to intercept every request.

The "Screen Select Mode" radio buttons allow you to specify which screen a clip view window will open upon. The "Frontmost if public" option means that windows will open on the frontmost screen if it's a public screen, and if it isn't a public screen the windows will fall back to opening on the default public screen. "Frontmost" is the dangerous option that will force clip view windows to open on the frontmost screen regardless of whether or not it is a public screen. This is in breach of the OS rules and general "inter-task courtesy" so don't be surprised if this option causes problems. Many non-public screens are simply not set up in a way to accept visitor windows. Lastly, "Default public" means that windows will always pop up on the default public screen.

The "Gadget Help" checkbox enables gadget help for clip view windows. Windows with gadget help turned on will provide a brief description of a gadget's function when the mouse is held over it and will tell you the type of data in a clipboard unit when the mouse is held over a unit. Windows without gadget help on have the virtue of taking up slightly less space, having a smaller minimum size, and perhaps being less annoying(?).

If you check the "Pop up" checkbox Clipper will pop up its control/prefs window whenever it is run. Most users will be launching Clipper whenever the Workbench is loaded by placing it in the WBStartup drawer, and will thus not want to select this option.

"View Screen" is the button to press to specify which screen clips will be shown to you on when you select one in a clip view window and then select the "view" option by gadget, keyboard or menu. See the Main Window section for more details.

Pressing "Show clips" is equivalent to pressing the clip view hotkey, and simply pops up a clip view window.

"Use" tells Clipper to use the current preferences settings but not to write them to disk. To make your preference settings permanent press the "Save" button instead. To cancel and changes you have made press the "Cancel" button.

The "About" button brings up the requester telling you lots of things you didn't want to know about me, Clipper and Haille.

Finally, pressing "Quit" will ask Clipper to get out of your face and un-do all the changes to the system that it has made.

Last words on the preferences window

All the gadgets have hot-keys that are underscored on the gadget labels and may be used instead of mouse clicks. In standard GadTools fashion pressing the tab key will cycle through all the integer gadgets.

Pressing the close window gadget or the escape key is equivalent to cancelling your changes.

## 1.10 Flames...

Bugs, Problems etc

First, the good news, that as far as I know this release version of Clipper at last has no serious bugs at all. It certainly has nothing that will crash the system or do any damage to anything else.

But, things aren't quite perfect and here's a list of some things you may notice...

1) Clipper Enforcer hits! Well yes it does, but this is absolutely not my fault and is in fact the doing of datatypes.library. That sounds like a bad excuse, but every other program that uses the datatypes system gets Enforcer hits in the same way, including Multiview etc. The bug occurs when datatypes.library creates datatypes objects from clip units 1-255 (unit 0 is fine.) As far as I can see this problem does no damage, at least not on my system. Very little I can do about this - perhaps v40 has the bug fixed?

2) Scrolling of datatypes (especially text ones) is unreliable i.e. sometimes you'll click on the datatype object and then drag and it won't scroll. The problem occurs more frequently with screens of greater depth. Again this isn't my fault and occurs with all programs using the datatypes system including Multiview.

3) Ignoring of mouse clicks. If a clip view window is NOT the selected window, and then you double click on a clip unit in order to try and direct a clip request (with the first click also activating the clip view window) the double click isn't registered. Similarly, if you just try and click once on a unit when the clip window isn't active, the click activates the window but is otherwise ignored. This is a result of the custom input handling I have to use to get around the clip unit datatypes objects stealing mouse presses, and there's nothing I can do to fix things.

4) Clipper misses a clipboard read or write. This should no longer happen, as Clipper now notices all clipboard requests (DoIO and SendIO type). However...

NOTE: A few (usually older) programs don't actually use the "real" OS clipboard system, although they offer facilities typical of the clipboard such as paste, copy etc. They use their own internal buffers to cut/copy to or paste from, and never make the data publicly available. Clipper will never be able to do anything about this sort of program. If you have a program that Clipper doesn't seem to be able to intercept, then paste something into the clipboard using the program and then summon up a clip view by hot key. If the data is present in the clipboard then Clipper did erroneously miss the clipboard write and you can report the bug to me. If the data is not in the clipboard, you've got hold of one of those few programs that have rather missed one of big plusses of the clipboard (data sharing) and are handling things themselves.

ALSO NOTE: If your application is really using the OS clipboard system, the only other possible reason for Clipper missing reads or writes is that the program opened the clipboard device before Clipper was run. This is extremely unlikely to happpen if Clipper is run during startup, but may possibly ocur if you run Clipper after an application has started.

Other than that, Clipper is just about as close as I can get it to problem free. Mungwall, and Enforcer both give it a clean bill of health (unlike datatypes.library) and after a LOT of testing the program seems stable.

## 1.11 How Clipper works...

How Clipper Works

This section is slightly technical and you certainly do not need to read it in order to use Clipper. It's here for those who are interested or those experiencing problems.

As soon as Clipper starts it performs some rather worrying operations - it patches Exec's DoIO, SendIO, OpenDevice and CloseDevice library calls. This is unfortunately necessary as there is no operating system supported method for patching device commands as you may do with library calls. Clipper then sits and keeps track of all tasks opening the clipboard device and builds its own lists of the tasks and the node they're using before passing the open device call on to the standard OS call. Do note that this means that any task that opens up the clipboard before Clipper is run (unlikely) will not be able to be intercepted by Clipper.

The Clipper DoIO and SendIO commands pick up on IO requests Clipper has previously intercepted at the time of opening the clipboard device. All sorts of games go on at this stage, and Clipper gets out of the way of all non-intercepted requests as quickly as possible. If the request is to be intercepted, then Clipper opens up a clip view window and waits for a user decision.

The clip view windows also have their fair share of worrying features. They each require some forbid (stop all other tasks) and permit calls to Exec library to guarantee that the clipboard openings made by datatypes.library don't in turn get intercepted by Clipper and cause a spiral headed for software failure with recursive openings of clip view windows to facilitate the opening of the first clip view window's clip unit views. Each window also installs its own high level input handler which is necessary to grab some mouse button input before it is eaten by datatypes objects. The input handlers have been written very efficiently (in assembler as is all of Clipper) and get out of the way of input they're not interested in as quickly as possible. To further complicate things, as all the Clipper code must be totally re-entrant a new input handler actually has to be built in real time, but be assured that Clipper carries out all the necessary cache clearing operations.

Once the user selects what to do with a clip request Clipper either sends the request through as normal if it was "passed through," emulates a failure with the most appropriate return code if "deny" was selected, or if a clip unit was selected by the user a difficult life is accepted and re-direction is undertaken. Here, Clipper opens up the clipboard device itself, and performs the read or write command asked for by the original IO request on the alternative unit, and then fills out the original IO request to reflect the results of the alternative read/write. Thus it looks to the application as if it is reading the clip unit it originally asked for. Unfortunately clip requests take place in a number of stages and Clipper has to spot old IO requests that it is sent back and in the process of re-directing and re-direct them each time it sees them. When the end of the clip read or write is detected Clipper also closes down its re-direction clipboard opening.

Close device is also patched so that Clipper can keep its list of clipboard openings up to date and free up its custom ClipOpen-Record that is opened for each request it knows about.

Getting windows to notice changes in clipboard contents is implemented in a somewhat non-standard way. Rather than using the clipboard's own notification of content changes (which is for a specific unit and would have involved installing 256 change handlers), a message is simply dispatched to all open clip view windows when Clipper notices a CMD_UPDATE.

And that is roughly how Clipper works. This is an attempt to do something that the OS does not strictly support and some of the programming involved is playing with fire, BUT I really have made every effort to write Clipper as well as possible and don't think it does anything wrong in itself. The code keeps very tight track of resources allocated and what other tasks are doing and in all the tests I've carried out it does seem impossible to confuse Clipper into doing anything dangerous. Needless to say you use Clipper at your own risk (see the disclaimer section ), but about the best vouch I can make for it is that we use it on our Amigas with lots of important source code and graphics flying around which we do not want lost. So far Clipper has failed to crash the system itself (although do watch out for programs with dodgy clipboard handling - we don't use any but I'm sure they exist.)

## 1.12 Things for the future

The Future

This version of Clipper was a rush, and thus does not have everything in that was planned. Most notably, the new drag and drop feature has been disabled. This allows you to drag the contents of a clip unit and drop it onto an applications window causes the application to paste in the contents of the dropped clip unit. Basically, very much like app icons.

The drag and drop system necessitated task specifc preferences, which are implemented although disabled for this release. You will be able to set things up differently for specific tasks in the future.

There are a number of other smaller things to add too, including support for the standard amiga c/v/x keys to make copying, moving and deleting clip units a little simpler.

All this and more will feature in the next version of Clipper due just after Easter.

## 1.13 Stuff...

Boring Bits

Copyright notice

Clipper is copyright ®1995/6/7 Roch Gadsdon of Vivid Software, all rights reserved. It is not in the public domain although you may copy and distribute it freely so long as the contents of the Clipper archive remain unchanged.

Software notice

Clipper is emperor-ware. Thus, if you use the program regularly you are required to find out how many years ago Haille Selassie died. It would also be appreciated if you'd e-mail or mail me just to say that you're using Clipper and perhaps that it's as useful as we hoped it might be. See the author section for contact details.

Clipper is certainly not shareware. IMHO, far, far too many projects are put out as shareware - who wants to pay money for an icon editor or the like?!

Disclaimer

You use Clipper at your own risk, and I cannot be held responsible for anything nasty it should do to you or your computer either directly or indirectly. I have made every effort to make Clipper safe, but it's your choice to use it and you accept any consequences. See the end of the workings section for more about reliability issues.

## 1.14 Clipper's author

Clipper's Author

Clipper was written by me, Roch Gadsdon, of the not so omnipotent if we're honest, Vivid Software. It was started after about five years of saying "Wouldn't it be good if...[most of Clipper's features]", and I think it does the job it was intended for very well. Clipper got written s-l-o-w-l-y... started at the end of Summer '95, left alone while I was back at university (Keele - it's a great place although too far North), had about a page of source added at Christmas, and was finally got ready for some sort of release at Easter. Unfortunately, bugs appeared about this time, so things got left again until a bored couple of days at the end of the 1996 Summer vacation.

Vivid Software are famous for two great demos: "Lame Talk" and "Abstrax Pig", and for the two games they half wrote and then, err, stopped ("Mr.Gurk's Amazing 16 Million Coloured Quest" and "Air Hockey.") The second of these is bloody good, but with our normal sense of timing we got it near to completion just as the Amiga games market collapsed.

And finally just a big thank to Chris for saying that Clipper was nice and generally being a cool person. The rest of the people who deserve thanking wisely have nothing to do with computers - except, perhaps, my brother Ben who might just read this, so a big thank to you too.

Feel free to contact me regarding bug reports, flames, comments etc...

E-mail: u3m46@keele.ac.uk [Valid university term time until June '97]

Snail mail: Roch Gadsdon

9 Herberton Road

Southbourne

BOURNEMOUTH

Dorset

BH6 5HU

England

## 1.15 Thanks

Thankyou and Goodnight

On a personal note, a big thank to Chris for saying that Clipper was nice and generally for being a cool person. And an equally big thank to my brother Ben for keeping things in perspective. Most of the rest of the people who deserve thanking wisely have nothing to do with computers. However, to those who have e-mailed with feedback and suggestions - thankyou, it's genuinely appreciated. All your ideas haven't made it in this time, but expect to see them in version 1.02 due at Easter.