Installation Instructions

The tar.Z file you just unpacked should contain a working Gopher binary, which you should copy to your Apps or LocalApps directory.

If you are upgrading from a previous version, please note that the defaults database name has changed, so you may have to re-enter your default server in the Preferences.

Gopher uses two other public domain packages, Stuart and Ph. Both should be availabe on popular archive sites (Ph is included here in binary form, source is available)

To build it, you may want to take a look at Parameters.h, and set the variables **DEFAULT_SERVER** and **DEFAULT_PORT** to your default

Gopher server (if you don't understand this, don't worry about it).

You will need to build the **play** command in its Play directory. Just cd into the Play directory and type "make". Then move the **play** program to its final destination (normally /usr/local/bin).

Also, change the variable **PlayCommand** if you don't want to install the play program in /usr/local/bin. You won't be able to see (and play) sounds if there is no play program.

Then just type "make", and you should have a Gopher executable. You may decide to strip it to conserve disk space (type *strip Gopher*).

History:

Version 1.12 - changes by Paul Lindner lindner@boombox.micro.umn.edu

Fixed the client so that it can work with the new index-server protocol change.

Upped the size of some data structures so that it can better deal with WAIS gateways.

Version 1.11 - changes by Paul Lindner lindner@boombox.micro.umn.edu

Made some cosemetic changes. Spiffed up some of the panels. Made them look pretty.

Added an Item Type Field in the Item info panel.

Fixed the Help menu item to better reflect reality.

Fixed up the highlighting of search terms. Also added a Find menu item so the user can find the relevant portions of the document, given the search string. (I still need to implement a find previous though.)

Version 1.1 - changes by David Lacey dlacey@umaxc.weeg.uiowa.edu

Added a Stuart Speaker interface so that Stuart could be messaged rather than just Terminal when a Telnet session was called. Also added a preferences item for this. Stuart is better than Terminal because, unlike terminal, Stuart doesn't start a new app each time you fire up a telnet session.

Added an alert box to show you the login to use when you do a telnet and a login was specified by the server.

Added a speaker to Rex Pruess's Ph program to handle CSO Namservers.

Fixed the initial menu negotiation -- Gopher used to send two \n\r sequences to the server causing it to die under some conditions (noticbly on a SLIP link)

Changed the window from 3 small panes to 2 larger panes.

Enlarged the Courier font used to represent the Telnet sessions -- better legiblity

Changed the Defaults database to look for "Gopher" rather than "UST_Gopher" to be more consistent with the NeXT defaults interface.

Added two new response code handlers:
        Type 'R' for rlogin. Useful because the system can negotiate the
login for the user transparently rather than having to give the user an alert
box telling them what to type.
        Type 'T' for tn3270 sessions. Useful for all those ugly blue boxes
that just won't go away...

        Changed default port from 150 to 70.

Gopher 1.0c release notes

Fixed an annoying problem with the index popup. Hitting the close button would
cause the program to hang.

Added a spiffy icon.

Gopher 1.0b2 release notes

Fixed a minor but annoying bug : I forgot to switch the default shell
for Terminal to what it was.

Gopher 1.0b release notes

This is the 1.0 release of my Gopher client. It is by no means perfect,
but I am leaving the country and I wanted to release something that
worked. The main changes from 0.9b :

- help text has been beefed up

- main window uses a SplitView so you can resize the Browser and the Text
- allows telnet connections (but in a rather ugly way).
- allows sounds to be played. To do so, however, you must first install the "play" program.

Possible problems :

When getting text from the server, the client just tries to read a large amount of data, under the assumption that the socket will be closed when there is no more text. This might not be true (i.e. the socket may not always be closed). The documentation of the protocol is not entirely clear on this subject.

Still relatively little error checking. I'm trying to keep it simple.