Info

Table of Contents

1 Introduction

$Id: info.tnf,v 1.17 1992/03/02 02:26:22 waite Exp $

Info is a program for reading documentation, which you are using now.

To learn how to use Info, type the command "h". It will bring you to a programmed instruction sequence.

The notation "C-x" (derived from that used in the GNU Emacs editor) is used to represent the ASCII Control-x character for any letter "x". "C-?" is used to represent the DELETE or RUBOUT character.

2 Beginners Help

2.1 Small Screen

Since your terminal has an unusually small number of lines on its screen, it is necessary to give you special advice at the beginning.

To move forward through the text and see another screen full, press the Space bar. To move back up, press the key labeled Rubout or Delete or DEL.

Here are 40 lines of junk, so you can try Spaces and Rubouts and see what they do. At the end are instructions of what you should do next.

This is line 17 This is line 18 This is line 19 This is line 20 This is line 21 This is line 22 This is line 23 This is line 24 This is line 25 This is line 26 This is line 27 This is line 28 This is line 29 This is line 30 This is line 31 This is line 32 This is line 33 This is line 34 This is line 35 This is line 36 This is line 37 This is line 38 This is line 39 This is line 40 This is line 41 This is line 42 This is line 43 This is line 44 This is line 45 This is line 46 This is line 47 This is line 48 This is line 49 This is line 50 This is line 51 This is line 52 This is line 53 This is line 54 This is line 55 This is line 56

If you have managed to get here, go back to the beginning with Rubout, and come back here again, then you understand Space and Rubout. So now type an "n"–just one character; don't type the quotes and don't type a Return afterward– to get to the normal start of the course.

2.2 Help

You are talking to the program Info, for reading documentation.

Right now you are looking at one "Node" of Information. A node contains text describing a specific topic at a specific level of detail. This node's topic is "how to use Info".

The top line of a node is its "header". This node's header (look at it now) says that it is the node named "Help" in the file "info". It says that the Next node after this one is the node called "Help-P". An advanced Info command lets you go to any node whose name you know.

Besides a "Next", a node can have a "Previous" or an "Up". This node has a "Previous" but no "Up", as you can see.

Now it's time to move on to the Next node, named "Help-P".

>> Type "n" to move there. Type just one character; don't type the quotes and don't type a Return afterward.

">>" in the margin means it is really time to try a command.

2.3 P

This node is called "Help-P". The "Previous" node, as you see, is "Help", which is the one you just came from using the "n" command. Another "n" command now would take you to the Next node, "Help-C-l".

>> But don't do that yet. First, try the "p" command, which takes you to the Previous node. When you get there, you can do an "n" again to return here.

This all probably seems insultingly simple so far, but DON'T be led into skimming. Things will get more complicated soon. Also, don't try a new command until you are told it's time to. Otherwise, you may make Info skip past an important warning that was coming up.

>> Now do an "n" to get to the node "Help-C-l" and learn more.

2.4 C-l

Space, Rubout, B and C-l commands.

This node's header tells you that you are now at node "Help-C-l", and that "P" would get you back to "Help-P". The line starting "Space," is a "Title", saying what the node is about (most nodes have titles).

This is a big node and it doesn't all fit on your display screen. The Space, Rubout (possibly called Delete on your terminal) and B commands exist to allow you to "move around" in a node that doesn't all fit on the screen at once. Space moves forward, to show what was below the bottom of the screen. Rubout moves backward, to show what was above the top of the screen (there isn't anything above the top until you have typed some spaces).

>> Now try typing a Space (afterward, type a Rubout to return here).

When you type the space, the two lines that were at the bottom of the screen appear at the top, followed by more lines. Rubout takes the two lines from the top and moves them to the bottom, USUALLY, but if there are not a full screen's worth of lines above them they may not make it all the way to the bottom.

If you type a Space when there is no more to see, it will ring the bell and otherwise do nothing. The same goes for a Rubout when the header of the node is visible.

If your screen is ever garbaged, you can tell Info to print it out again by typing C-l (Control-l, that is–hold down "Control" and type an "L" or "l").

>> Type C-l now.

To move back to the beginning of the node you are on, you can type a lot of Rubouts. You can also type simply "b" for beginning.

>> Try that now. (I have put in enough verbiage to try and make sure you are not on the first screenful now). Then come back, with Spaces.

You have just learned a considerable number of commands. If you want to use one but have trouble remembering which, you should type a "?" which will print out a brief list of commands. When you are finished looking at the list, make it go away by typing a Space.

>> Type a "?" now. After it finishes, type a Space.

From now on, you will encounter large nodes without warning, and will be expected to know how to use Space and Rubout to move around in them without being told. Since not all terminals have the same size screen, it would be impossible to warn you anyway.

>> Now type "n" to see the description of the "m" command.

2.5 Menus and the M command

With only the "n" and "p" commands for moving between nodes, nodes are restricted to a linear sequence. Menus allow a branching structure. A menu is a list of other nodes you can move to. These nodes may be any nodes in the documentation but are usually subnodes of the current node. A menu actually just part of the text of the node formatted specially so that Info can interpret it. The beginning of a menu is always identified by a line which starts with "* Menu:". A node contains a menu if and only if it has a line in it which starts that way. The only menu you can use at any moment is the one in the node you are in. To use a menu in any other node, you must move to that node first.

After the start of the menu, each line that starts with a "*" identifies one subtopic. The line will usually contain a brief name for the subtopic (followed by a ":"), the name of the node that talks about that subtopic, and optionally some further description of the subtopic. Lines in the menu that don't start with a "*" have no special meaning - they are only for the human reader's benefit and do not define additional subtopics. Here is an example:

```
* Foo:  FOO's Node This tells about FOO
```

The subtopic name is Foo, and the node describing it is "FOO's Node". The rest of the line is just for the reader's Information. [[But this line is not a real menu item, simply because there is no line above it which starts with "* Menu:".]]

When you use a menu to go to another node (in a way that will be described soon), what you specify is the subtopic name, the first thing in the menu line. Info uses it to find the menu line, extracts the node name from it, and goes to that node. The reason that there is both a subtopic name and a node name is that the node name must be meaningful to the computer and may therefore have to be ugly looking. The subtopic name can be chosen just to be convenient for the user to specify. Often the node name is convenient for the user to specify and so both it and the subtopic name are the same. There is an abbreviation for this:

```
* Foo::   This tells about FOO
```

Here the node name has been left out (represented by the two colons). By default the node name will be the same as the subtopic name. Thus in this case the subtopic name and node name are the same; they are both "Foo".

>> Now use Spaces to find the menu in this node, then come back to the front with a "b". As

you see, a menu is actually visible in its node. If you can't find a menu in a node by looking at it, then the node doesn't have a menu and the "m" command is not available.

The command to go to one of the subnodes is "m" - but DON'T DO IT YET! Before you use "m", you must understand the difference between commands and arguments. So far, you have learned several commands that do not need arguments. When you type one, Info processes it and is instantly ready for another command. The "m" command is different: it is incomplete without the NAME OF THE SUBTOPIC. Once you have typed "m", Info tries to read the subtopic name.

Now look for the line containing many dashes near the bottom of the screen. There is one more line beneath that one, but usually it is blank. If it is empty, Info is ready for a command, such as "n" or "b" or Space or "m". If that line contains text ending in a colon, it mean Info is trying to read the ARGUMENT to a command. At such times, commands won't work, because Info will try to use them as the argument. You must either type the argument and finish the command you started, or type C-g to cancel the command. When you have done one of those things, the line will become blank again.

The command to go to a subnode via a menu is "m". After you type the "m", the line at the bottom of the screen says "Menu item: ". You must then type the name of type subtopic you want, and end it with a Return.

You can abbreviate the subtopic name. If the abbreviation is not unique, the first matching subtopic is chosen. Some menus will put the shortest possible abbreviation for each subtopic name in capital letters, so you can see how much you need to type. It does not matter whether you use upper case or lower case when you type the subtopic. You should not put any spaces at the end, or inside of the item name, except for one space where a space appears in the item in the menu.

Here is a menu to give you a chance to practice.

>> Now type just an "m" and see what happens:

Now you are "inside" an "m" command. Commands can't be used now; the next thing you will type must be the name of a subtopic.

You can change your mind about doing the "m" by typing C-g.

>> Try that now; notice the bottom line clear.

>> Then type another "m".

>> Now type "BAR", the item name. Don't type Return yet.

While you are typing the item name, you can use the Rubout character to cancel one character at a time if you make a mistake.

>> Type one to cancel the "R". You could type another "R" to replace it. You don't have to, since "BA" is a valid abbreviation.

>> Now you are ready to go. Type a Return.

After visiting Help-FOO, you should return here.

>> Type "n" to see more commands.

2.6 The U command

Congratulations! This is the node Help-FOO. Unlike the other nodes you have seen, this one has an "Up": "Help-M", the node you just came from via the "m" command. This is the usual convention– the nodes you reach from a menu have Ups that lead back to the menu. Menus move Down in the tree, and Up moves Up. Previous, on the other hand, is usually used to "stay on the same level but go backwards".

You can go back to the node Help-M by typing the command "u" for "Up". That will put you at the FRONT of the node - to get back to where you were reading you will have to type some Spaces.

>> Now type "u" to move back up to Help-M.

2.7 Some advanced Info commands

The course is almost over, so please stick with it to the end.

If you have been moving around to different nodes and wish to retrace your steps, the "l" command ("l" for "last") will do that, one node at a time. If you have been following directions, an "l" command now will get you back to Help-M. Another "l" command would undo the "u" and get you back to Help-FOO. Another "l" would undo the M and get you back to Help-M.

>> Try typing three "l"'s, pausing in between to see what each "l" does. Then follow directions again and you will end up back here.

Note the difference between "l" and "p": "l" moves to where YOU last were, whereas "p" always moves to the node which the header says is the "Previous" node (from this node, to Help-M).

The "d" command gets you instantly to the Directory node. This node, which is the first one you saw when you entered Info, has a menu which leads (directly, or indirectly through other menus), to all the nodes that exist.

>> Try doing a "d", then do an "l" to return here (yes, DO return).

Sometimes, in Info documentation, you will see a cross reference. Cross references look like this: See Section 2.8 [Help-Cross], page 10. That is a real, live cross reference which is named "Cross" and points at the node named "Help-Cross".

If you wish to follow a cross reference, you must use the "f" command. The "f" must be followed by the cross reference name (in this case, "Cross"). You can use Rubout to edit the name, and if you change your mind about following any reference you can use C-g to cancel the command.

Completion is available in the "f" command; you can complete among all the cross reference names in the current node.

>> Type "f", followed by "Cross", and a Return.

To get a list of all the cross references in the current node, you can type "?" after an "f". The "f" continues to await a cross reference name even after printing the list, so if you don't actually want to follow a reference you should type a C-g to cancel the "f".

>> Type "f?" to get a list of the footnotes in this node. Then type a C-g and see how the "f" gives up.

>> Now type "n" to see the last node of the course.

2.8 Cross

This is the node reached by the cross reference named "Cross".

While this node is specifically intended to be reached by a cross reference, most cross references lead to nodes that "belong" someplace else far away in the structure of Info. So you can't expect the footnote to have a Next, Previous or Up pointing back to where you came from. In general, the "l" (el) command is the only way to get back there.

>> Type "l" to return to the node where the cross reference was.

2.9 Q

To get out of Info, back to what you were doing before, type "q" for "Quit".

This is the end of the course on using Info. There are some other commands that are not essential or meant for experienced users; they are useful, and you can find them by looking in the directory for documentation on Info. Finding them will be a good exercise in using Info in the usual manner.

>> Type "d" to go to the Info directory node; then type "mInfo" and Return, to get to the node about Info and see what other help is available.

2.10 Some Advanced Info Commands

If you know a node's name, you can go there by typing "g", the name, and Return. Thus, "gTop<Return>" would go to the node called Top in this file (its directory node). "gExpert<Return>" would come back here.

Unlike "m", "g" does not allow the use of abbreviations.

To go to a node in another file, you can include the filename in the node name by putting it at the front, in parentheses. Thus, "g(dir)Top<Return>" would go to the Info Directory node, which is node Top in the file dir.

The node name "*" specifies the whole file. So you can look at all of the current file by typing "g*<Return>" or all of any other file with "g(FILENAME)<Return>".

The "s" command allows you to search a whole file for a string. It will switch to the next node if and when that is necessary. You type "s" followed by the string to search for, terminated by Return. To search for the same string again, just "s" followed by Return will do. The file's nodes will be scanned in the order they are in in the file, which has no necessary relationship to the order that they may be in in the tree structure of menus and next's. But normally the two orders will not be far different. In any case, you can always do a "b" to find out what node you have reached, if the header isn't visible (this can happen, because "S" puts your cursor at the occurrence of the string, not at the beginning of the node).

If you grudge the system each character of type-in it requires, you might like to use the commands "1", "2", "3", "4", and "5". They are short for the "m" command together with an argument. "1" goes through the first item in the current node's menu; "2" goes through the second item, etc. Note that numbers larger than 5 are not allowed. If the item you want is that far down, you are better off using an abbreviation for its name than counting.

2.11 Printing things from inside Info

For information on how Info decides which printer to use for these commands, see Section 2.12 [Options], page 11.

It is possible to use Info to print two types of information. The first is just any information that you can display on the screen. Using the C-t command, you can print the contents of the current node to a printer. Note that the printout that you get will be just a straight "ASCII" dump of the node with no formatting other than that that you see on the screen.

For many applications of Info, it is also possible to get high-quality laser printed copies of manuals which are being viewed in Info. The C-p command has the effect of trying to print the current MANUAL (ie. not just the current node) as a POSTSCRIPT document. You should check with your system maintainer whether this sort of printing is supported for your documents. Also be careful what you print as the laser printer versions of many documents can be quite large.

2.12 Info Options.

It is possible for you to customize much of Info's behaviour. In particular, you can customize which printer is used by the printing commands (see Section 2.11 [Printing], page 11). The "o" command is used to examine and/or set Info options using an interactive interface. These options are local to your Info session and have no effect on any shell variables or other variables you may have outside this session. The Info options can be initialized by setting the values of shell environment variables (see example below) but their values do not propagate back to those shell variables.

When you execute the "o" command, a display such as the following will appear at the top of your screen.

```
The options which can be set are:

EDITOR              /usr/ucb/vi
    Program to use for editing (see also WINEDITOR).
INFORUNDISPLAY      (null)
    Program to use to display output of 'R' command in parallel.
WINEDITOR           (null)
    Program to use for editing in parallel (preferred over EDITOR).
PRINTER             lp
    Printer to send any printouts to.
```

This display lists the currently available options, their current values, and a description of each option. At the bottom of the screen you will be asked the question:

```
Set which option (RET to finish):
```

To this you type the name of the option which you want to change, or a RETURN if you are finished with the options. Then you will see the prompt:

```
To what:
```

to which you type the desired value. (Eg. "shelltool vi" for the option WINEDITOR.) The screen will be updated to reflect the new value which will be in effect from then on.

If you don't want to have to set the options each time you enter Info, you can use environment variables. The Info options are initialised by the values (if any) of the environment variables with the same names. For example, to make the printer "myprinter" the default printer for Info, execute the following command in csh(1):

```
setenv PRINTER myprinter
```

If the environment variables EDITOR and PRINTER are not set, then the Info options EDITOR and PRINTER default to "/usr/ucb/vi" and "lp", respectively, as shown above.

2.13 Completing Input

To decrease the amount that you have to type Info supports completion. This means that in certain contexts where Info has a good idea what you might type, it can help you. One such context is when you are using the "m" command to select a node from a menu. Info provides facilities so that you only have to specify the unique part of the thing you are typing so that it can be distinguished from the other menu items.

The following characters cause special behaviour when completing:

SPACE, C-t, TAB
: Add to what has been typed so far as much as can be inferred. If this is not unique, ring the bell.

?
: Print at the top of the screen a list of the completions of what has been typed so far, but don't add anything.

RETURN, NEWLINE
: Attempt to complete what has been typed so far into a unique value. If this is possible accept that value as the input value, otherwise complete as much as possible and ring the bell.

As a simple example, consider a menu containing the items "One", "Two" and "Only". When executing the "m" command to select something from this menu, you could type "T" then RETURN to select "Two", and "O", TAB, "e", RETURN to select "One". Note that because "One" and "Only" share a common prefix you must type the "e" to enable Info to distinguish the two. In this case completion doesn't save you anything. In practice, node names for example, may be quite long and completion is very useful.

Other things that can be accessed using completion are cross references ("f" command), options ("o" command) and file names ("e" command, see Section 2.14 [Examples], page 13).

2.14 Interacting with examples using Info

Info provides a powerful facility for interacting with examples. It enables authors to build interactive examples into the documentation tree and thus make them part of their system's documentation. Two facilities are provided for users: running and editing. These are flagged by the words "Edit" and/or "Run" in the header of a node. Eg.

```
File: doc1,  Node: top,  Next: another,  Eg: EditRun
```

When "Run" is present, the "r" command is enabled. This command invokes a set of author-specified commands to run the example associated with the current node. Once these commands are finished, possibly after significant interaction with the user, a transcript of the results is displayed.

If the Info option INFORUNDISPLAY is set, it will be assumed to be the name of a program to use to display the transcript of an Info run. It should take one filename argument. For example, users of SunView may use the value "textedit" to get the output of the run displayed in another window.

If INFORUNDISPLAY is not set, the transcript is displayed by info as a node, so that it may be viewed using the normal Info commands. In this case the "v" command can be used to return to the run output at a later stage in the Info session.

Usually associated with each example provided using Info are one or more files. Typically these files will reside in a documentation directory and are maintained as part of the system they illustrate. Info provides a facility for allowing you to modify these files using the "e" command. To do this, Info arranges for each file that you edit to be copied into your directory (ie. the one in which you ran Info). Thus you are editing your own copy of the file, not the system copy. (Note that the "r" command will also copy the files into your directory in order to be able to run the example commands.)

The "e" command will prompt you for the name of the file to edit. Since completion is available, you can use the "?" character to determine which files are available for editing along with a small description of each file. Once you have selected the file to edit, an editor is invoked to do the editing. The options EDITOR and WINEDITOR are assumed to be a program which takes one argument and edits the file. If WINEDITOR is set it will be executed in the background, and is normally used when you are in a windowing environment like SunView (eg. with a value of "shelltool vi"). If WINEDITOR is not set, EDITOR will be used. Note that EDITOR will default to "/usr/ucb/vi" (see Section 2.12 [Options], page 11).

If you use WINEDITOR you will be able to edit and use Info at the same time (assuming you have a window system). You can exit WINEDITOR at any time you like. If you use EDITOR you will have to finish editing before you can resume your Info session.

Using the "r" and "e" commands, it is possible for you to try out supplied examples, modify them, try out the modified examples, and so on. In addition, you obtain your own copies of the files comprising the example for possible use outside of Info.

2.15 Complaining about things

Working under the principle that the best time to make a complaint is when the complaint is fresh in your mind, Info provdes the "c" command. This enables you to easily send mail to the maintainer of the Info system (and/or the system that is documented by the Info system).

The "c" command simply invokes your mail sender and lets you type your message in ended with C-d. The message is sent to the appropriate person.

Note that most mail senders will let you abort sending a message using your interrupt character (usually either C-c or Delete). If you decide not to complain after all, use that method to abort and return to your Info session.

2.16 Standalone use of Info

Info may be used inside GNU Emacs, but there is also a standalone version (called infoReader in the Eli system). This section briefly lists the options which may be used when running Info in this standalone manner.

Synopsis

```
info [ -d info-directory-path ] [ -f info-file ]
  [ -o dump-file] [ -n node-name ] [ menu-name ... ]
```

Description

When Info is invoked, the optional *menu-name* arguments are the series of menu options that you wish to follow from the main menu. For example, to get info about using buffers in Emacs, you might type 'info emacs buffers' (assuming that the Emacs information is available).

The options are as follows:

`-d` A quick way for you to temporarily change the path searched by Info when it starts up. You can also set the environment variable `INFO_DIRECTORY_PATHS` to a list of colon-separated directory names.

`-f` Specifies which Info file to load. Normally, Info looks for the file '`dir`' in the info search path, and goes to node `Top` in that file. The '`dir`' file contains a menu whose items all point to info files. If the info file you wish to view is not in that menu, you can say `info -f info-filename`, and that file specifically will be used.

`-n` Sets the initial node to look for. This allows you to read an info file that doesn't have a `Top` node.

`-o` Dumps the contents of the node that you have specified, so that you can do with it what you please. Note that you can print the node that you are currently looking at with `C-t` from within Info, so it isn't clear what you want to do with the output from `-o`.

2.17 Internals

Info files are ASCII files with a particular structure. Here we discuss that structure while describing the make-up of the major components of an Info file.

Note that much of the discussion below assumes a context where GNU Emacs is being used. (In particular to manipulate the Info file and for validation.)

2.17.1 Add

To add a new topic to the list in the directory, you must 1) create a node, in some file, to document that topic. 2) put that topic in the menu in the directory. *Note Menu: Menus.

The new node can live in an existing documentation file, or in a new one. It must have a C-_ character before it (invisible to the user; this node has one but you can't see it), and it ends with either a C-_, a C-l, or the end of file. Note: If you put in a C-l to end a new node, be sure that there is a C-_ after it to start the next one, since C-l can't START a node. Also, a nicer way to make a node boundary be a page boundary as well is to put a C-l RIGHT AFTER the C-_.

The C-_ starting a node must be followed by a newline or a C-l newline, after which comes the node's header line. The header line must give the node's name (by which Info will find it), and state the names of the Next, Previous, and Up nodes (if there are any). As you can see, this node's Up node is the node Top, which points at all the documentation for Info. The Next node is "Menus".

The keywords "Node", "Previous", "Up" and "Next", may appear in any order, anywhere in the header line, but the recommended order is the one in this sentence. Each keyword must be followed by a colon, spaces and tabs, and then the appropriate name. The name may be terminated with a tab, a comma, or a newline. A space does not end it; node names may contain spaces. The case of letters in the names is insignificant.

A node name has two forms. A node in the current file is named by what appears after the "Node: " in that node's first line. For example, this node's name is "Add". A node in another file is named by "(FILENAME)NODE-WITHIN-FILE", as in "(info)Add" for this node. If the file name is relative, it is taken starting from the standard Info file directory of your site. The name "(FILENAME)Top" can be abbreviated to just "(FILENAME)". By convention, the name "Top" is used for the "highest" node in any single file - the node whose "Up" points out of the file. The Directory node is "(dir)". The Top node of a document file listed in the Directory should have an "Up: (dir)" in it.

The node name "*" is special: it refers to the entire file. Thus, g* will show you the whole current file. The use of the node * is to make it possible to make old-fashioned, unstructured files into nodes of the tree.

The "Node:" name, in which a node states its own name, must not contain a filename, since Info when searching for a node does not expect one to be there. The Next, Previous and Up names may contain them. In this node, since the Up node is in the same file, it was not necessary to use one.

Note that the nodes in this file have a File name in the header line. The File names are ignored by Info, but they serve as comments to help identify the node for the user.

2.17.2 Menus

Any node in the Info hierarchy may have a MENU–a list of subnodes. The "m" command searches the current node's menu for the topic which it reads from the terminal.

A menu begins with a line starting with "* Menu:". The rest of the line is a comment. After the starting line, every line that begins with a "* " lists a single topic. The name of the topic–the arg that the user must give to the "m" command to select this topic– comes right after the star and space, and is followed by a colon, spaces and tabs, and the name of the node which discusses that topic. The node name, like node names following Next, Previous and Up, may be terminated with a tab, comma, or newline; it may also be terminated with a period.

If the node name and topic name are the same, than rather than giving the name twice, the abbreviation "* NAME::" may be used (and should be used, whenever possible, as it reduces the visual clutter in the menu).

It is considerate to choose the topic names so that they differ from each other very near the beginning–this allows the user to type short abbreviations. In a long menu, it is a good idea to capitalize the beginning of each item name which is the minimum acceptable abbreviation for it (a long menu is more than 5 or so entries).

The node's listed in a node's menu are called its "subnodes", and it is their "superior". They should each have an "Up:" pointing at the superior. It is often useful to arrange all or most of the subnodes in a sequence of Next's/Previous's so that someone who wants to see them all need not keep revisiting the Menu.

The Info Directory is simply the menu of the node "(dir)Top"–that is, node Top in file .../info/dir. You can put new entries in that menu just like any other menu. The Info Directory is NOT the same as the file directory called "info". It happens that many of Info's files live on that file directory, but they don't have to; and files on that directory are not automatically listed in the Info Directory node.

Also, although the Info node graph is claimed to be a "hierarchy", in fact it can be ANY directed graph. Shared structures and pointer cycles are perfectly possible, and can be used if they are appropriate to the meaning to be expressed. There is no need for all the nodes in a file to form a connected structure. In fact, this file has two connected components. You are in one of them, which is under the node Top; the other contains the node Help which the "h" command goes to. In fact, since there is no garbage collector, nothing terrible happens if a substructure is not pointed to, but such a substructure will be rather useless since nobody will ever find out that it exists.

2.17.3 Cross References

A cross reference can be placed anywhere in the text, unlike a menu item which must go at the front of a line. A cross reference looks like a menu item except that it has "*note" instead of "*". It CANNOT be terminated by a ")", because ")"'s are so often part of node names. If you wish to enclose a cross reference in parentheses, terminate it with a period first. Here are two examples of cross references pointers:

*Note details: Menus. (See *note 3: Internals.)

2.17.4 Tag Tables

You can speed up the access to nodes of a large Info file by giving it a tag table. Unlike the tag table for a program, the tag table for an Info file lives inside the file itself and will automatically be used whenever Info reads in the file.

To make a tag table, go to a node in the file using Info and type M-x Info-tagify. Then you must use C-x C-s to save the file.

Once the Info file has a tag table, you must make certain it is up to date. If, as a result of deletion of text, any node moves back more than a thousand characters in the file from the position recorded in the tag table, Info will no longer be able to find that node. To update the tag table, use the Info-tagify command again.

An Info file tag table appears at the end of the file and looks like this:

```
C-_C-l
Tag Table:
File: info, Node: Cross-refsC-?21419
File: info,  Node: TagsC-?22145
C-_
End Tag Table
```

Note that it contains one line per node, and this line contains the beginning of the node's header (ending just after the node name), a rubout character, and the character position in the file of the beginning of the node.

2.17.5 Checking an Info File

When creating an Info file, it is easy to forget the name of a node when you are making a pointer to it from another node. If you put in the wrong name for a node, this will not be detected until someone tries to go through the pointer using Info. Verification of the Info file is an automatic process which checks all pointers to nodes and reports any pointers which are invalid. Every Next, Previous, and Up is checked, as is every menu item and every cross reference. In addition, any Next which doesn't have a Previous pointing back is reported. Only pointers within the file are checked, because checking pointers to other files would be terribly slow. But those are usually few.

To check an Info file, do M-x Info-validate while looking at any node of the file with Info.

Index