makeguide

COLLABORATORS			
	TITLE : makeguide		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 17, 2023	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

makeguide

1.1 makeguide.guide

```
                makeinfo
********

   This chapter documents the use of the makeinfo program, versions
1.55 and later. Parts of it are extracted from the Texinfo manual.
```


1.2 makeguide.guide/Requirements

Requirements
****

Makeinfo 1.55 runs only with version 2.04 or higher of the operating system.

In order to print the documentation on a printer you need TeX, the typesetting language developed by Donald E. Knuth. If you want to use texinfotimes.tex you need the Postscript fonts and a TeX able to handle Postscript fonts. AmigaTeX by Radical Eye Software handles PostScript fonts correctly.

In order to view the AmigaGuide(R) files, you need to copy the amigaguide.library from the libs directory in libs:.

1.3 makeguide.guide/Distribution

Distribution
****

Makeguide is distributed under terms of the GNU copying license. Please read the file COPYING contained in this distribution.

AmigaGuide, AmigaGuide.info, amigaguide.library, WDisplay, WDisplay.info (C) Copyright 1991-93 Commodore-Amiga, Inc. All Rights Reserved. Reproduced and distributed under license from Commodore.

AMIGAGUIDE SOFTWARE IS PROVIDED AS IS AND SUBJECT TO CHANGE; NO WARRANTIES ARE MADE. ALL USE IS AT YOU OWN RISK. NO LIABILITY OR RESPONSIBILITY IS ASSUMED.

1.4 makeguide.guide/Why makeinfo?

Why makeinfo?
****

1.5 makeguide.guide/How it started

How it started
==============

This project started with the idea of bringing to the Amiga community a set of tools which would have greatly simplified the handling of on-line help as opposed to printed manuals. Currently, the Amiga programmer has to keep two separate files which have to be updated in parallel. If also some hypertext feature is desired, the document writer has to keep a third version of the file.

This is unbearable. So we began to look around to find a solution of this problem. We finally found something that would be sufficient for this tasks: it's called Texinfo.

1.6 makeguide.guide/What is Texinfo?

What is Texinfo?
================

The GNU people have found a great way of working around the problem described in the previous section.

A Texinfo document is written in a very simple dialect of TeX that is easy to learn and use, and it's specifically tailored for the creation of technical manuals. Texinfo focuses on logical aspects--so the @t{} command, which typesets in fixed width font whatever is in the braces, should be never used, and rather replaced with @code{}, This also ensures that each user can customize its Texinfo macros in such a way to spot out specific parts of a Texinfo file, or to set a different page size, text formatting etc (an example is texinfotimes.tex that prints .texinfo with PostScript fonts).

Of course, the format has to be rich enough to express all the needs of a technical manual, and small enough to allow a decent translation of all the available constructs to plain ASCII (for an hypotetical hypertext viewer). Also in this respect Texinfo is an excellent balance.

Full documentation is available on how to write a Texinfo document. It is written, of course, in Texinfo, and is very clear. You should be able to start authoring a Texinfo document in an hour or so. If you're used to TeX, ten minutes will suffice. This documentation can be found on most ftp sites which have GNU stuff. The latest version of Texinfo we know of is 3.0. It's a final, very stable version and the differences to the previous version (2.16) are very small (indeed GNU identifies the version inside the documentation as 2.18 but the release is called 3.0).

1.7 makeguide.guide/Output modes

Output modes

Once a Texinfo document is ready, it can be converted in one of the following ways:

1. It can be passed through TeX using the set of macros texinfo.tex. You will get a beautiful printed manual.

2. It can be passed through the standard makeinfo in its Info mode; it will generate an Info document (Info is the GNU hypertext reader). This is not much of interest for you.

3. It can be passed through makeinfo with the -no-headers option; it will generate an almost plain ASCII text file (cross references and indices will still be in Info style).

   But with our version of makeinfo,

4. It can be passed through makeinfo with the -amiga option; it will generate an AmigaGuide(R) document; cross references, menus and indices will become buttons.

5. It can be passed through makeinfo with the -amiga -no-headers options; it will generate a plain ASCII text file.

6. It can be passed through makeinfo with the -amiga-39 option; it will generate an AmigaGuide(R) V39 document with bold and italics attributes; cross references, menus and indices will become buttons.

7. It can be passed through makeinfo with the -amiga-39 -no-headers options; it will generate a ASCII text file with ANSI escape sequences for boldface and italics.

See
Amiga specific options
, for more details on the Amiga-specific features of makeinfo.

Thus, you have to maintain just a document in order to produce a printed manual, on-line hypertext documentation and plain ASCII documentation. Texinfo of course relies on a series of information you specify, like menus, cross references, etc., but the amount of work you'll need to do this will be very small. It also makes a great job in converting to ASCII--for instance, @emph{} text, which comes out slanted in the manual, is shown with two asterisk around, the new option -amiga-39 produces boldface for AmigaGuide(R) 3.0. We would like to urge all of the Amiga community programmers, in particular the people distributing documentation as a file, to start using Texinfo. It will be an enormous quantum leap.

Since makeguide was first released (July 1992), some Amiga developers started to write the documentation in Texinfo; we know of:

* ToolManager by Stefan Becker;

* Term by Olaf Barthel;

* F-Finder by Markus Juani Aalto;

* Cerca by Carlo Todeschini;

* Arcalc by Roberto Attias;

* SuperDuper, Mostra, Leggi, ne (the nice editor) by Sebastiano Vigna;

* Most by Uwe Röhm;

* Assigns 1.1 by Enrico Fedrigo;

* MegaD 3.0 (soon to be released);

* Makeguide 1.55 by Reinhard Spisser and Sebastiano Vigna 8^).

In addition to this, you can have near all the GNU documentation in AmigaGuide(R) format (more than 15MB of .texi files).

1.8 makeguide.guide/What is makeinfo

What is makeinfo?
****

This chapter documents the use of the makeinfo program, version 1.55. It is extracted from the Texinfo manual (with minor modifications).

makeinfo is a program for converting Texinfo files into Info and AmigaGuide(R) files. Texinfo is a documentation system that uses a single source file to produce both on-line information and printed output.

See the Texinfo manual, to learn about the Texinfo documentation system.

Formatting Control

Options

Pointer Validation

1.9 makeguide.guide/Formatting Control

```
Controlling Paragraph Formats
=============================

   In general, makeinfo fills the paragraphs that it outputs to an Info
file.  Filling is the process of breaking and connecting lines so that
lines are the same length as or shorter than the number specified as
the fill column.  Lines are broken between words.  With makeinfo, you
can control:

   * The width of each paragraph (the fill-column).

   * The amount of indentation that the first line of each paragraph
     receives (the paragraph-indentation).
```

1.10 makeguide.guide/Options

```
                Command Line Options
====================

   The following command line options are available for makeinfo.

-D var
     Cause var to be defined.  This is equivalent to  @set var in the
     Texinfo file.

-error-limit limit
     Set the maximum number of errors that makeinfo will report before
     exiting (on the assumption that continuing would be useless).  The
     default number of errors that can be reported before makeinfo
     gives up is 100.

-fill-column width
     Specify the maximum number of columns in a line; this is the
     right-hand edge of a line.  Paragraphs that are filled will be
     filled to this width.  The default value for fill-column is 72.

-footnote-style style
     Set the footnote style to style, either end for the end node style
     or separate for the separate node style.  The value set by this
     option overrides the value set in a Texinfo file by an
     @footnotestyle command.  When the footnote style is separate,
     makeinfo makes a new node containing the footnotes found in the
     current node.  When the footnote style is end, makeinfo places the
     footnote references at the end of the current node.

-I dir
     Add dir to the directory search list for finding files that are
     included using the  @include command.  By default, makeinfo
     searches only the current directory.

-no-headers
```

Do not include menus or node lines in the output. This results in an ASCII file that you cannot read in Info since it does not contain the requisite nodes or menus; but you can print such a file in a single, typewriter-like font and produce acceptable output.

-no-split
Suppress the splitting stage of makeinfo. Normally, large output files (where the size is greater than 70k bytes) are split into smaller subfiles, each one approximately 50k bytes. If you specify -no-split, makeinfo will not split up the output file.

-no-pointer-validate
-no-validate
Suppress the pointer-validation phase of makeinfo. Normally, after a Texinfo file is processed, some consistency checks are made to ensure that cross references can be resolved, etc. See

Pointer Validation
.

-no-warn
Suppress the output of warning messages. This does not suppress the output of error messages, only warnings. You might want this if the file you are creating has examples of Texinfo cross references within it, and the nodes that are referenced do not actually exist.

-no-number-footnotes
Supress automatic footnote numbering. By default, makeinfo numbers each footnote sequentially in a single node, resetting the current footnote number to 1 at the start of each node.

-output file
-o file
Specify that the output should be directed to file and not to the file name specified in the @setfilename command found in the Texinfo source. file can be the special token -, which specifies standard output.

-paragraph-indent indent
Set the paragraph indentation style to indent. The value set by this option overrides the value set in a Texinfo file by an @paragraphindent command. The value of indent is interpreted as follows:

* If the value of indent is asis, do not change the existing indentation at the starts of paragraphs.

* If the value of indent is zero, delete any existing indentation.

* If the value of indent is greater than zero, indent each paragraph by that number of spaces.

-reference-limit limit
Set the value of the number of references to a node that makeinfo

```
     will make without reporting a warning.  If a node has more than
     this number of references in it, makeinfo will make the references
     but also report a warning.

-U var
     Cause var to be undefined.  This is equivalent to  @clear var in
     the Texinfo file.

-verbose
     Cause makeinfo to display messages saying what it is doing.
     Normally, makeinfo only outputs messages if there are errors or
     warnings.

-version
     Report the version number of this copy of makeinfo.

-amiga
     Convert the Texinfo file in a AmigaGuide(R) hypertext file.

-amiga-39
     Convert the Texinfo file in a AmigaGuide(R) V39 hypertext file,
     with support for italic, boldface and other options. If
     -no-headers is also specified, makeinfo produces an ASCII file
     with escape sequences for italic and boldface. See

                Amiga specific options
                .
```

1.11 makeguide.guide/Pointer Validation

```
Pointer Validation
==================

   If you do not suppress pointer-validation (by using the
-no-pointer-validation option), makeinfo will check the validity of the
final Info file.  Mostly, this means ensuring that nodes you have
referenced really exist.  Here is a complete list of what is checked:

  1. If a 'Next', 'Previous', or 'Up' node reference is a reference to a
     node in the current file and is not an external reference such as
     to (dir), then the referenced node must exist.

  2. In every node, if the 'Previous' node is different from the 'Up'
     node, then the 'Previous' node must also be pointed to by a 'Next'
     node.

  3. Every node except the 'Top' node must have an 'Up' pointer.

  4. The node referenced by an 'Up' pointer must contain a reference to
     the current node in some manner other than through a 'Next'
     reference.  This includes menu entries and cross references.

  5. If the 'Next' reference of a node is not the same as the 'Next'
```

reference of the ‘Up’ reference, then the node referenced by the ‘Next’ pointer must have a ‘Previous’ pointer that points back to the current node. This rule allows the last node in a section to point to the first node of the next chapter.

1.12 makeguide.guide/Amiga specific options

```
                Amiga specific options
**********************

   This chapter describes the Amiga specific options that were added to
makeinfo. See
                Why makeinfo?
                , how these features were implemented.


                 Amiga output

                 Localization of strings

                 Index button

                 Conversion of spaces in node names

                 Using Postscript fonts
```

1.13 makeguide.guide/Amiga output

```
Amiga output
============

   Two options have been added to the GNU makeinfo:

   -amiga which converts to AmigaGuide(R) V34

   -amiga-39 which converts to AmigaGuide(R) V39 (using bold and
italic).

   When the -amiga and -no-headers options are used, a plain ASCII file
will be produced.

   When the -amiga-39 and -no-headers options are used, makeinfo
generates ANSI escape control sequences for bold and italics.
```

1.14 makeguide.guide/Localization of strings

Localization of strings

Starting with release 2.1 the Amiga Operating system supports localization of the user interface. Until now it was not possible to localize the hard-coded strings in makeinfo (you had to re-compile makeinfo) and texinfo.tex. Recompiling makeinfo needs at least 3.5 MB of free RAM space, the GNU C compiler (makeinfo uses a few GNU C 2.0 features), and a hard disk. Compiling makeinfo on a 68030 took about 10 minutes (the source is one file over 200KB long). Probably someone didn't even like the built-in strings and wanted to replace them. This was a big handicap for people writing documentation in different languages than English and wanted to write the documentation in makeinfo.

makeinfo (and texinfo.tex) now supports localization for all hardcoded strings. An easy to use way was provided to do this task.

The localization is done by setting one or more of the following variables in your .Texinfo file:

```
@set variable string to translate
```

where variable could be one of the following:

xrefstring
: Replace the standard see string that makeinfo and texinfo.tex prints in front of a @pxref{} and @ref{} with the string given as argument.

Xrefstring
: Replace the standard See string that makeinfo and texinfo.tex prints in front of a @xref{} with the string given as argument.

Footnotestring
: Replace the standard Footnote string that makeinfo prints before starting to print the footnotes. This variable has only effect for makeinfo.

Chapterstring
: Replace the standard Chapter string that texinfo.tex use for page headings, cross references, and others. This variable has only effect for texinfo.tex.

Appendixstring
: Replace the standard Appendix string that texinfo.tex use for page headings, cross references, and others. This variable has only effect for texinfo.tex.

Sectionstring
: Replace the standard Section string that texinfo.tex use for page headings, cross references, and others. This variable has only effect for texinfo.tex.

sectionstring
: Replace the standard section string that texinfo.tex use for page

```
     headings, cross references, and others.This variable has only
     effect for texinfo.tex. (this is the section string with the
     starting s lowercase).

pagestring
     Replace  the standard page string that texinfo.tex use for page
     headings, cross references, and others. This variable has only
     effect for texinfo.tex.

   Note: To undefine the variables, you can't  @set them with no
arguments; if you do so, you set the translation string to an empty
string. You must use  @unset variable to undefine the variable.

   An example:

   To get a different string printed before the cross reference, you
should add a  @set command in your Texinfo file:

     @set xrefstring vedi
     @set Xrefstring Vedi

   or you can add a define at the makeinfo command line:

   makeinfo -amiga-39 -D "xrefstring vedi" -D "Xrefstring Vedi" foo.texi

   Note: The localization of the strings xrefstring and Xrefstring has
only effect if you set both variables.

   You may ask: why to implement the localization as variables and not
as new options? We decided to do so for the following reasons:

   * we wanted to limit the attitional options of makeinfo with respect
     to the original GNU makeinfo to a minimal number (in effect only
     for the various versions of AmigaGuide(R));

   * a second reason is that the implementation of localization via
     options to would have required many more work (it took us about 15
     minutes to see how to implement it and about 1 minute to implement
     it);

   * a nice side-effect of this is that you can re-set the strings
     within the document, allowing you to make a unique multi-language
     document;

   * since the localization features were also implemented in the macro
     packages texinfo.tex and texinfotimes.tex, you can use this
     feature also when printing the .Texinfo file (you can't do this if
     you specify the translated string as a command line argument of
     makeinfo). For this reason you should  @set the translations
     in the .Texinfo file;

   * you can specify the translation strings in the .Texinfo file
     without worrying if the string refers only to texinfo.tex or
     makeinfo.

   * in other words, we decided to do so because doing it with variables
     made the localization easier to use and more powerful.
```

```
   Example:

      @set xrefstring vedi
      @set Xrefstring Vedi
      @xref{Index button,},

   results in

   Vedi
                Index button
                ,

      @set xrefstring Give a nice look at
      @set Xrefstring Give a proper look at
      @xref{Index button,},

   results in

   Give a proper look at
                Index button
                ,
```

Remember! You have to specify both xrefstring and Xrefstring if you want to translate the strings that are printed before a cross reference. @set-ting only xrefstring or Xrefstring does have no effect.

Note: It's more useful if you @set the localization variables in the Texinfo document rather than using the -D feature of makeinfo, because you can also have the same effect on the printed Texinfo files.

You can @set the variable more than one time in your document; doing so you can have pieces of code having different strings. (this because the status of the variables is verified when a cross-reference is written). This could be very useful when you're writing only one Texinfo file containing the documentation in different languages.

1.15 makeguide.guide/Index button

```
Index button
============
```

The Index button points to the Concept Index (if the Concept Index exists). This is done by inserting a @Index Concept Index immediatly after the beginning of the Concept Index node.

1.16 makeguide.guide/Conversion of spaces in node names

```
Conversion of spaces in node names
==================================
```

Since AmigaGuide(R) does not support spaces in node names (when trying to get an online help with SendAmigaGuideCmdA()), the spaces are converted in underscores if the variable amiga_convert_nodes is set (done with @set amiga_convert_nodes in your Texinfo file).

As with the localization feature of makeinfo, you can @set the conversion within the document, you have the same advantages as with the localization variables.

1.17 makeguide.guide/Using Postscript fonts

```
Using Postscript fonts
======================
```

Since we have a TeX on the Amiga that handles correctly and without any problem the Adobe Postscript fonts, we decided to make a variation of the texinfo.tex macro package that prints the Texinfo files with these fonts. Computer Modern roman, italic, oblique and bold is substituted by Times roman, Times italic, Times oblique and Times bold. The Computer Modern TypeWriter font is substituted by Courier, the sans-serif style is substituted by Helvetica (actually Texinfo does not use the sans-serif style). The result of this is placed in the file texinfotimes.tex.

To print a Texinfo document with Times fonts, the only thing you have to do is to substitute the \input texinfo line at the beginning of the Texinfo file with \input texinfotimes. You you need to recompile the file with TeX. After recompiling it you can use TeX's previewer to look how nice it is or use the printer drivers to print it.

1.18 makeguide.guide/Differences from GNU makeinfo

```
                Differences from GNU makeinfo
*****************************
```

While we were working on the project, it became clear that much more was possible with this new tool. For instance, all of the GNU software comes with Texinfo documentation. It was a matter of a couple of minutes to make the Texinfo manual (540K!) into an AmigaGuide(R) file that we could browse at will, in full AmigaGuide(R) hypertext.

Every port of this kind has of course some kind of tradeoff. Info and AmigaGuide(R) have a rather vaste intersection of features, but both have specific features that can't be mimicked on the other side.

The main design choice at the specification level was that every

Texinfo should have been convertible to an AmigaGuide(R) file. And that every Texinfo file prepared for AmigaGuide(R) should have been convertible to Info format (this is essential, e.g., for people doing cross-development).

Fortunately, we were enough lucky: all Info features are translatable in AmigaGuide(R) features, except for file splitting; but AmigaGuide(R) doesn't load a file entirely--just loads a part of it. So even a 1 megabyte document is not a problem for it. Info instead relies on a splitting mechanism which is in makeinfo: this mechanism is disabled by the options -amiga and -amiga-39.

There are also a couple of problems with the representation of buttons; Info does not have real buttons--it just put in the text something of the form *Note Nodename::. Thus, the aspect of a line in Info or AmigaGuide(R) is very different. We chose to put in the button name only the real title of the button, i.e., the destination node name if nothing else is specified, or the second argument of a @ref, @xref or @pxref command if available. Moreover, we noted that the absence of *Note's had made the text a bit unpleasant to read; so we inserted see and See whenever Texinfo would have done it in the printed text. See

Localization of strings
, for more detailed information on how localize these strings.

For menus, Texinfo relies on the visualisation of both the button name and the destination node name when building indices. Cutting away the destination node name would have resulted in a horribly looking index. So we decided to preserve the visualization of the destination node name after a menu button when building an index. A normal menu will instead show just a button.

Also, we slightly improved an aspect of @pxref. Since no ending point was needed before the closing parenthesis, we stripped it off.

All the features of Info which maps to AmigaGuide(R) have been implemented. Thus, the Top node of a Texinfo document will become the Main node of the AmigaGuide(R) file, the @master command will be set to the name of the Texinfo file, and the @width command will be set to the value makeinfo is using for formatting text. The Index button points to the Concept Index (if the Concept Index exists).

The -no-headers option has now a better behaviour when also -amiga is selected. *Note's are replaced by see, etc.. The result is a completely human readable ASCII file. -no-headers in combination with -amiga-39 generates ASCII files with ANSI escape sequences. See

Localization of strings
, for more detailed information on how localize these strings.

1.19 makeguide.guide/Texindex

```
Texindex
********

   This chapter explains how to use texindex. It is extraced from the
Texinfo manual (with small modifications).

   The tex formatting commands itself does not sort the indices; it
writes an output file of unsorted index data.  This is a misfeature of
TeX.  Hence, to generate a printed index, you first need a sorted index
to work from.  The texindex command sorts indices.  (The source file
texindex.c comes as part of this distribution.)

     Usage: texindex [-k] [-T tempdir] infile [-o outfile] ...

   Each infile arg can optionally be followed by a -o outfile arg; for
each infile that is not followed by a -o arg, the infile name with s
(for 'sorted') appended is used for the outfile.

   -T dir is the directory to put temp files in, instead of /tmp.

   -k means 'keep tempfiles', for debugging.

   The tex formatting command outputs unsorted index files under names
that obey a standard convention.  These names are the name of your main
input file to the tex formatting command, with everything after the
first period thrown away, and the two letter names of indices added at
the end.  For example, the raw index output files for the input file
foo.texinfo would be foo.cp, foo.vr, foo.fn, foo.tp, foo.pg and foo.ky.
Those are exactly the arguments to give to texindex.

   For each file specified, texindex generates a sorted index file
whose name is made by appending s to the input file name.  The
does not alter the raw index output file.

   After you have sorted the indices, you need to rerun the tex
formatting command on the Texinfo file.  This regenerates a formatted
DVI file with up-to-date index entries.(1)

   To summarize, this is a three step process:

  1. Run the tex formatting command on the Texinfo file.  This
     generates the formatted DVI file as well as the raw index files
     with two letter extensions.

  2. Run the shell command texindex on the raw index files to sort
     them.  This creates the corresponding sorted index files.

  3. Rerun the tex formatting command on the Texinfo file.  This
     regenerates a formatted DVI file with the index entries in the
     correct order.  This second run also corrects the page numbers for
     the cross references.  (The tables of contents are always correct.)

   You need not run texindex each time after you run the tex
formatting.  If you do not, on the next run, the tex formatting command
```

will use whatever sorted index files happen to exist from the previous use of texindex. This is usually OK while you are debugging.

---------- Footnotes ----------

(1) If you use more than one index and have cross references to an index other than the first, you must run tex three times to get correct output: once to generate raw index data; again (after texindex) to output the text of the indices and determine their true page numbers; and a third time to output correct page numbers in cross references to them. However, cross references to indices are rare.

1.20 makeguide.guide/Known Problems & Bugs

Known Problems & Bugs
****

1.21 makeguide.guide/Problems

Problems
========

It should be noted that unfortunately Texinfo doesn't have any way of specifying a line in a node. So this feature of AmigaGuide(R) is not representable in a Texinfo file. This is indeed the biggest drawback of using Texinfo for writing AmigaGuide(R) documents. But we feel that the advantages are definitely overwhelming.

Due to the fact that AmigaGuide(R) is rather picky about the characters it allows in a button name or in a node name, at least for the time being makeinfo has to replace in a node name every slash or backslash with a dash and every quotes with a single open quote. Still, names with curly braces in them will behave incorrectly. But this is an AmigaGuide(R) problem.

Note: Texinfo doesn't support officially extended or accented characters. The workaround, for the time being, is to use ASCII extended characters like à, etc. in the text (in place of the various \`, etc. commands). makeinfo will process the document flawlessly, and by including the amigatexinfo.tex file supplied in this distribution archive before texinfo.tex into your document, Texinfo will typeset correctly the accented characters (this should work on any TeX system which supports TeX 3.0; we tried successfully AmigaTeX and PasTeX). Note also that the first (October) release of makeinfo 1.49 had an

incorrect amiga.tex file, which did not work with many foreign characters.

1.22 makeguide.guide/Bugs

Bugs
====

Finally, we want to describe some bugs we found both in Texinfo and AmigaGuide(R), whose knowledge can be helpful.

AmigaGuide(R) V39 does not handle correctly the @ character. When AmigaGuide(R)(R) finds this character in the first column of a line, it simply ignores the entire word (this because its commands always have to begin in the first column of a line). Due to implementation problems, we didn't solve this problem (sorry). AmigaGuide(R) 39 ignores the word also if @ is the first character in an attribute. So @section is ignored. makeinfo puts a space character before it, so that the commands are showed. This is only one of the reasons why we decided to not support the @wordwrap feature of AmigaGuide(R) V39 (with wordwrap the possibility to get a @ command at the first column is high, and even this commands are not displayed). Since the wordwrap of AmigaGuide(R) 39 is rather simple, everything different than plain text would appeared very bad.

AmigaGuide(R) 33.1369 doesn't work properly if the name of a button contains an apostrophe ('). So you shouldn't use button names containing @code{}, @samp{}, etc. (they are always forbidden in node names, but you can specify a button name different from the node name in an external reference command). The workaround is to not use apostrophes in button names (AmigaGuide(R) V34 and V39 doesn't suffer from this problem). Since now AmigaGuide(R) V34 is freely distributable, you should have no problem upgrading to the new versions. The makeguide distribution archive contains the latest version of AmigaGuide(R) V34 (version 34.8, dated 3-May-93).

Texinfo 2.16 and 3.0 will insert an unnamed TOC entry for the Top node when the automatic pointer generation feature of makeinfo is used (that is, your Top node is defined via @node Top followed by @top). You should include the whole Top node in a @ifinfo/@end ifinfo pair for the time being.

texinfo.tex version 2.108 has wrong settings for @afourpaper. We corrected these problems in the version 2.109 (included with this archive).

makeinfo does not sort correctly the index if an entry contains formatting commands, like @code. Index entries should be written without formatting commands or making different entries for Info and Texinfo. The Index entries of makeguide.texi are written in this way.

1.23 makeguide.guide/Distribution Files

```
                Distribution Files
******************

   The archive mkguide154.lha contains the following files:

   In the root directory:

makeinfo
     The makeinfo program. See
                What is makeinfo
                , for more information
     how to use it.

texindex
     Used for sorting the Indexes generated by Texinfo (when compiling
     it with TeX). You need to run TeX twice to have the Indexes
     updated. See
                Texindex
                , for more information how to use it.

COPYING
     The GNU distribution license. makeguide is distributed under this
     license. Please read it!

CHANGES
     Changes made to makeinfo since the latest release.

README
     Read it!

AmigaGuide
     Utility that displays AmigaGuide(R) documents. You need
     amigaguide.library in your libs: directory.

   In the libs directory:

amigaguide.library
     This library contains functions to handle hyper-text documents.
     You should copy it in your libs: directory in order to work. Use
     the AmigaGuide utility to display the hypertext documents (e.g.
     makeguide.guide in the doc directory.

   In the src directory:

getopt.c
     Support file for makeinfo.

getopt1.c
     Support file for makeinfo.

getopt.h
     Support file for makeinfo.

makeinfo.c
```

```
     The source for makeinfo.

texindex.c
     The source for texindex.

Makefile
     Makefile to build makeinfo. You need the GNU C compiler to compile
     it (makeinfo.c uses some features specific to the GNU C compiler).
     The makefile has options to compile it with Markus Wild's GCC port
     (actual version 2.3.3) and Davide Pasetto's GCC port (actual
     version 2.2.2).

   In the texinfo directory:

   This directory contains the macros for use with TeX.

texinfo.tex
     The macro package that defines the language of Texinfo. You should
     copy this file in a path where TeX can find it. Version 2.109.

texinfotimes.tex
     Same as texinfo.tex, but with font descriptions for PostScript. You
     need the Postscript fonts Times, Helvetica and Courier installed
     and a TeX able to handle with PostScript fonts. Version 2.109.

amigatexinfo.tex
     Special file for converting the accents in specific TeX
     characters. If you use accents in your .texinfo file, you should
     include this file before texinfo.tex or texinfotimes.tex

   In the doc directory:

Makeguide.texi
     The documentation of makeinfo in Texinfo format. You can use
     makeinfo program to convert it to plain ASCII file or to the
     AmigaGuide(R) hypertext format. See
                Options
                , to learn more about
     how to generate output. You can also compile this document with
     texinfo.tex or texinfotimes.tex to have a nice output on the
     printer.

Makeguide.guide
     The documentation of makeinfo in AmigaGuide(R) V39 hypertext
     format.  Note that AmigaGuide(R) V34 displays this file without
     problems, since all unrecognized commands and attributes are
     simply ignored by AmigaGuide(R).

Makeguide.doc
     The documentation of makeinfo in ASCII with ANSI escape sequences.
     You need a file-viewer able to handle ANSI escape sequences (like
     Leggi).

Makeguide.dvi
     The Device Independent format. You can print it on your printer
     (with TeX).
```

```
Makeguide.ps
     The documentation in PostScript format. If you print it on a
     PostScript printer (or on a non-PostScript printer using post),
     you can see the beautiful output generated by texinfotimes.tex.

Makefile
     Makefile for generating the various documentation types from the
     documentation source file makeguide.texi.
```

1.24 makeguide.guide/Acknowledgments

```
Acknowledgments
***************

   We would like to thank GNU for the creation of this great
documentation system and Tom Rokicki for allowing us to distribute the
amigatexinfo.tex file, which is based on the amiga.tex file supplied in
the distribution of Radical Eye's AmigaTeX, and for his invaluable
assistance in adapting amiga.tex to Texinfo.

   Stefan Becker made some very useful suggestions; most of them are now
implemented in this version of makeinfo; he was also a beta-tester for
this new version.
```

1.25 makeguide.guide/Author Info

```
Author Info
***********

   Good Texinfo-ing! Please send any bug report or enhancement request
to

         Reinhard Spisser
         Via Licurgo 12
         I-20162 Milano MI

         BIX: reinhards@bix.com
         INTERNET: spisser@ghost.sm.dsi.unimi.it
                   rein@bliss.st.dsi.unimi.it


         Sebastiano Vigna
         Via California 22
         I-20144 Milano MI

         BIX: svigna@bix.com
         INTERNET: vigna@ghost.dsi.unimi.it
         UUCP: seba@sebamiga.adsp.sub.org
```

1.26 makeguide.guide/Concept Index

Concept Index
****