

**gnu:guide/makeinfo**

**COLLABORATORS**

	<i>TITLE :</i> gnu:guide/makeinfo		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		July 22, 2024	

**REVISION HISTORY**

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>gnu:guide/makeinfo</b>	<b>1</b>
1.1	gnu:guide/makeinfo.guide . . . . .	1
1.2	makeinfo.guide/What is makeinfo . . . . .	1
1.3	makeinfo.guide/Formatting Control . . . . .	1
1.4	makeinfo.guide/Options . . . . .	2
1.5	makeinfo.guide/Pointer Validation . . . . .	4

---

# Chapter 1

## gnu:guide/makeinfo

### 1.1 gnu:guide/makeinfo.guide

```
`makeinfo'  
*****
```

This file documents the use of the `makeinfo` program, versions 1.51 and later. It is an extract from the `Texinfo` manual.

What is makeinfo

### 1.2 makeinfo.guide/What is makeinfo

```
What is `makeinfo`?  
*****
```

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

You can read the on-line information using Info; type `info` to learn about Info. See Texinfo, to learn about the Texinfo documentation system.

Formatting Control  
Options  
Pointer Validation

### 1.3 makeinfo.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").

## 1.4 makeinfo.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`

Suppress 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'`.

## 1.5 makeinfo.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.