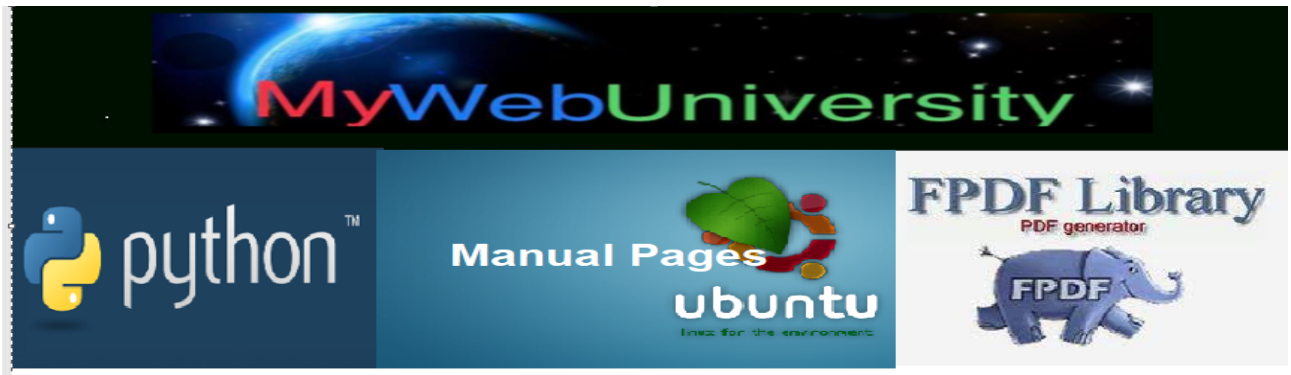

Full credit is given to the above companies including the OS that this PDF file was generated!

Linux Ubuntu 26.04 Manual Pages on command 'readelf.1'

$ man readelf.1

READELF(1) GNU Development Tools READELF(1)

NAME

readelf - display information about ELF files

SYNOPSIS

readelf [-a|--all]

[-h|--file-header]

[-l|--program-headers|--segments]

[-S|--section-headers|--sections]

[-g|--section-groups]

[-t|--section-details]

[-e|--headers]

[-s|--syms|--symbols]

[--dyn-syms|--lto-syms]

[--sym-base=[0|8|10|16]]

[--demangle=style|--no-demangle]

[--quiet]

[--recurse-limit|--no-recurse-limit]

[-U method|--unicode=method]

[-X|--extra-sym-info|--no-extra-sym-info]

[-n|--notes]

[-r|--relocs]

[-u|--unwind]

[-d|--dynamic]

[-V|--version-info]

[-A|--arch-specific]

[-D|--use-dynamic]

[-L|--lint|--enable-checks]

[-x <number or name>|--hex-dump=<number or name>]

[-p <number or name>|--string-dump=<number or name>]

[-R <number or name>|--relocated-dump=<number or name>]

[-j <number or name>|--display-section=<number or name>]

[-z|--decompress]

[-c|--archive-index]

[-w[lLiaprmfFsoORtUuTgAck]|

--debug-dump[=rawline,=decodedline,=info,=abbrev,=pubnames,=aranges,=macro,=frames,=frames-interp,=str,=str-offsets,=loc,=Ranges,=pubtypes,=trace_info,=trace_abbrev,=trace_aranges,=gdb_index,=addr,=cu_index,=links]]

[-wK|--debug-dump=follow-links]

[-wN|--debug-dump=no-follow-links]

[-wD|--debug-dump=use-debuginfod]

[-wE|--debug-dump=do-not-use-debuginfod]

[-P|--process-links]

[--dwarf-depth=n]

[--dwarf-start=n]

[--ctf=section]

[--ctf-parent=section]

[--ctf-symbols=section]

[--ctf-strings=section]

[--sframe=section]

[-I|--histogram]

[--got-contents]

[-v|--version]

[-W|--wide]

[-T|--silent-truncation]

[-H|--help]

elffile...

DESCRIPTION

readelf displays information about one or more ELF format object files. The options control what particular information to display.

elffile... are the object files to be examined. 32-bit and 64-bit ELF files are supported, as are archives containing ELF files.

This program performs a similar function to objdump but it goes into more detail and it exists independently of the BFD library, so if there is a bug in BFD then readelf will not be affected.

OPTIONS

The long and short forms of options, shown here as alternatives, are equivalent. At least one option besides -v or -H must be given.

-a

--all

Equivalent to specifying --file-header, --program-headers, --sections, --symbols, --relocs, --dynamic, --notes, --got-contents, --version-info, --arch-specific, --unwind, --section-groups and --histogram.

Note - this option does not enable --use-dynamic itself, so if that option is not present on the command line then dynamic symbols and dynamic relocs will not be displayed.

-h

--file-header

Displays the information contained in the ELF header at the start of the file.

-l

--program-headers

--segments

Displays the information contained in the file's segment headers, if it has any.

--quiet

Suppress "no symbols" diagnostic.

-S

--sections

--section-headers

Displays the information contained in the file's section headers, if it has any.

-g

--section-groups

Displays the information contained in the file's section groups, if it has any.

-t

--section-details

Displays the detailed section information. Implies -S.

-s

--symbols

--syms

Displays the entries in symbol table section of the file, if it has one. If a symbol has version information associated with it then this is displayed as well. The version string is displayed as a suffix to the symbol name, preceded by an @ character. For example foo@VER_1. If the version is the default version to be used when resolving unversioned references to the symbol then it is displayed as a suffix preceded by two @ characters. For example foo@@VER_2.

--dyn-syms

Displays the entries in dynamic symbol table section of the file, if it has one. The output format is the same as the format used by the --syms option.

--lto-syms

Displays the contents of any LTO symbol tables in the file.

--sym-base=[0|8|10|16]

Forces the size field of the symbol table to use the given base. Any unrecognized options will be treated as 0. --sym-base=0 represents the default and legacy behaviour. This will output sizes as decimal for numbers less than 100000. For sizes 100000 and greater hexadecimal notation will be used with a 0x prefix. --sym-base=8 will give the symbol sizes in octal. --sym-base=10 will always give the symbol sizes in decimal. --sym-base=16 will always give the symbol sizes in hexadecimal with a 0x prefix.

-C

--demangle[=style]

Decode (demangle) low-level symbol names into user-level names. This makes C++ function names readable. Different compilers have different mangling styles. The optional demangling style argument can be used to choose an appropriate demangling style for your compiler.

--no-demangle

Do not demangle low-level symbol names. This is the default.

--recurse-limit

--no-recurse-limit

--recursion-limit

--no-recursion-limit

Enables or disables a limit on the amount of recursion performed whilst demangling strings. Since the name mangling formats allow for an infinite level of recursion it is possible to create strings whose decoding will exhaust the amount of stack space available on the host machine, triggering a memory fault. The limit tries to prevent this from happening by restricting recursion to 2048 levels of nesting.

The default is for this limit to be enabled, but disabling it may be necessary in order to demangle truly complicated names. Note however that if the recursion limit is disabled then stack exhaustion is possible and any bug reports about such an event will be rejected.

-U [d|i|l|e|x|h]

--unicode=[default|invalid|locale|escape|hex|highlight]

Controls the display of non-ASCII characters in identifier names. The default (--unicode=locale or --unicode=default) is to treat them as multibyte characters and display them in the current locale. All other versions of this option treat the bytes as UTF-8 encoded values and attempt to interpret them. If they cannot be interpreted or if the --unicode=invalid option is used then they are displayed as a sequence of hex bytes, encloses in curly parethesis characters.

Using the --unicode=escape option will display the characters as as unicode escape sequences (\uxxxx). Using the --unicode=hex will display the characters as hex byte sequences enclosed between angle brackets.

Using the --unicode=highlight will display the characters as unicode escape sequences but it will also highlighted them in red, assuming that colouring is supported by the output device. The colouring is intended to draw attention to the presence of unicode sequences when they might not be expected.

-X

--extra-sym-info

When displaying details of symbols, include extra information not normally presented. Currently this just adds the name of the section referenced by the symbol's index field,

if there is one. In the future more information may be displayed when this option is enabled.

Enabling this option effectively enables the --wide option as well, at least when displaying symbol information.

--no-extra-sym-info

Disables the effect of the --extra-sym-info option. This is the default.

-e

--headers

Display all the headers in the file. Equivalent to -h -l -S.

-n

--notes

Displays the contents of the NOTE segments and/or sections, if any.

-r

--relocs

Displays the contents of the file's relocation section, if it has one.

-u

--unwind

Displays the contents of the file's unwind section, if it has one. Only the unwind sections for IA64 ELF files, as well as ARM unwind tables (".ARM.exidx" / ".ARM.extab") are currently supported. If support is not yet implemented for your architecture you could try dumping the contents of the .eh_frames section using the --debug-dump=frames or --debug-dump=frames-interp options.

-d

--dynamic

Displays the contents of the file's dynamic section, if it has one.

-V

--version-info

Displays the contents of the version sections in the file, if they exist.

-A

--arch-specific

Displays architecture-specific information in the file, if there is any.

-D

--use-dynamic

When displaying symbols, this option makes readelf use the symbol hash tables in the file's dynamic section, rather than the symbol table sections.

When displaying relocations, this option makes readelf display the dynamic relocations rather than the static relocations.

-L

--lint

--enable-checks

Displays warning messages about possible problems with the file(s) being examined. If used on its own then all of the contents of the file(s) will be examined. If used with one of the dumping options then the warning messages will only be produced for the things being displayed.

-x <number or name>

--hex-dump=<number or name>

Displays the contents of the indicated section as a hexadecimal bytes. A number identifies a particular section by index in the section table; any other string identifies all sections with that name in the object file. This option can be repeated multiple times on the command line in order to request multiple hex dumps.

-R <number or name>

--relocated-dump=<number or name>

Displays the contents of the indicated section as a hexadecimal bytes. A number identifies a particular section by index in the section table; any other string identifies all sections with that name in the object file. The contents of the section will be relocated before they are displayed. This option can be repeated multiple times on the command line in order to request multiple relocated dumps.

-p <number or name>

--string-dump=<number or name>

Displays the contents of the indicated section as printable strings. A number identifies a particular section by index in the section table; any other string identifies all sections with that name in the object file. This option can be repeated multiple times on the command line in order to request multiple string dumps.

-j <number or name>

--display-section

Displays the contents of the indicated section according to its section header type.

Sections containing relocations will be displayed as if the --relocations option had been used, sections contains symbols will be displayed as if the --syms option had been used and so on.

A number identifies a particular section by index in the section table; any other string identifies all sections with that name in the input file(s).

This option can be repeated multiple times on the command line in order to request multiple section dumps.

-z

--decompress

Requests that the section(s) being dumped by x, R or p options are decompressed before being displayed. If the section(s) are not compressed then they are displayed as is.

-c

--archive-index

Displays the file symbol index information contained in the header part of binary archives. Performs the same function as the t command to ar, but without using the BFD library.

-w[lLiaprmfFsOoRtUuTgAckK]

--debug-dump[=rawline,=decodedline,=info,=abbrev,=pubnames,=aranges,=macro,=frames,=frames-interp,=str,=str-offsets,=loc,=Ranges,=pubtypes,=trace_info,=trace_abbrev,=trace_aranges,=gdb_index,=addr,=cu_index,=links,=follow-links]

Displays the contents of the DWARF debug sections in the file, if any are present.

Compressed debug sections are automatically decompressed (temporarily) before they are displayed. If one or more of the optional letters or words follows the switch then only those type(s) of data will be dumped. The letters and words refer to the following information:

"a"

"=abbrev"

Displays the contents of the .debug_abbrev section.

"A"

"=addr"

Displays the contents of the .debug_addr section.

"c"

"=cu_index"

Displays the contents of the .debug_cu_index and/or .debug_tu_index sections.

"f"

"=frames"

Display the raw contents of a .debug_frame section.

"F"

"=frames-interp"

Display the interpreted contents of a .debug_frame section.

"g"

"=gdb_index"

Displays the contents of the .gdb_index and/or .debug_names sections.

"i"

"=info"

Displays the contents of the .debug_info section. Note: the output from this option can also be restricted by the use of the --dwarf-depth and --dwarf-start options.

"k"

"=links"

Displays the contents of the .gnu_debuglink, .gnu_debugaltlink and .debug_sup sections, if any of them are present. Also displays any links to separate dwarf object files (dwo), if they are specified by the DW_AT_GNU_dwo_name or DW_AT_dwo_name attributes in the .debug_info section.

"K"

"=follow-links"

Display the contents of any selected debug sections that are found in linked, separate debug info file(s). This can result in multiple versions of the same debug section being displayed if it exists in more than one file.

In addition, when displaying DWARF attributes, if a form is found that references the separate debug info file, then the referenced contents will also be displayed.

Note - in some distributions this option is enabled by default. It can be disabled via the N debug option. The default can be chosen when configuring the binutils via the --enable-follow-debug-links=yes or --enable-follow-debug-links=no options. If these are not used then the default is to enable the following of debug links.

Note - if support for the debuginfod protocol was enabled when the binutils were built then this option will also include an attempt to contact any debuginfod

servers mentioned in the DEBUGINFOD_URLS environment variable. This could take some time to resolve. This behaviour can be disabled via the =do-not-use-debuginfod debug option.

"N"

"=no-follow-links"

Disables the following of links to separate debug info files.

"D"

"=use-debuginfod"

Enables contacting debuginfod servers if there is a need to follow debug links. This is the default behaviour.

"E"

"=do-not-use-debuginfod"

Disables contacting debuginfod servers when there is a need to follow debug links.

"l"

"=rawline"

Displays the contents of the .debug_line section in a raw format.

"L"

"=decodedline"

Displays the interpreted contents of the .debug_line section.

"m"

"=macro"

Displays the contents of the .debug_macro and/or .debug_macinfo sections.

"o"

"=loc"

Displays the contents of the .debug_loc and/or .debug_loclists sections.

"O"

"=str-offsets"

Displays the contents of the .debug_str_offsets section.

"p"

"=pubnames"

Displays the contents of the .debug_pubnames and/or .debug_gnu_pubnames sections.

"r"

"=aranges"

Displays the contents of the .debug_aranges section.

"R"

"=Ranges"

Displays the contents of the .debug_ranges and/or .debug_rnglists sections.

"s"

"=str"

Displays the contents of the .debug_str, .debug_line_str and/or .debug_str_offsets sections.

"t"

"=pubtype"

Displays the contents of the .debug_pubtypes and/or .debug_gnu_pubtypes sections.

"T"

"=trace_aranges"

Displays the contents of the .trace_aranges section.

"u"

"=trace_abbrev"

Displays the contents of the .trace_abbrev section.

"U"

"=trace_info"

Displays the contents of the .trace_info section.

Note: displaying the contents of .debug_static_funcs, .debug_static_vars and debug_weaknames sections is not currently supported.

--dwarf-depth=n

Limit the dump of the ".debug_info" section to n children. This is only useful with --debug-dump=info. The default is to print all DIEs; the special value 0 for n will also have this effect.

With a non-zero value for n, DIEs at or deeper than n levels will not be printed. The range for n is zero-based.

--dwarf-start=n

Print only DIEs beginning with the DIE numbered n. This is only useful with --debug-dump=info.

If specified, this option will suppress printing of any header information and all DIEs before the DIE numbered n. Only siblings and children of the specified DIE will be

printed.

This can be used in conjunction with --dwarf-depth.

-P

--process-links

Display the contents of non-debug sections found in separate debuginfo files that are linked to the main file. This option automatically implies the -wK option, and only sections requested by other command line options will be displayed.

--ctf[=section]

Display the contents of the specified CTF section. CTF sections themselves contain many subsections, all of which are displayed in order.

By default, display the name of the section named .ctf, which is the name emitted by ld.

--ctf-parent=member

If the CTF section contains ambiguously-defined types, it will consist of an archive of many CTF dictionaries, all inheriting from one dictionary containing unambiguous types. This member is by default named .ctf, like the section containing it, but it is possible to change this name using the "ctf_link_set_memb_name_changer" function at link time. When looking at CTF archives that have been created by a linker that uses the name changer to rename the parent archive member, --ctf-parent can be used to specify the name used for the parent.

--ctf-parent-section=section

This option lets you pick a completely different section for the CTF parent dictionary containing unambiguous types than for the child dictionaries that contain the ambiguous remainder. The linker does not emit ELF objects structured like this, but some third-party linkers may. It's also convenient to inspect CTF written out as multiple raw files to compose them with objcopy, which can put them in different ELF sections but not in different members of a single CTF dict.

--ctf-symbols=section

--ctf-strings=section

Specify the name of another section from which the CTF file can inherit strings and symbols. By default, the ".symtab" and its linked string table are used.

If either of --ctf-symbols or --ctf-strings is specified, the other must be specified as well.

--got-contents

Displays the contents of the file's Global Offset Table (GOT) sections, if it has any.

For MIPS, this option is similar to --arch-specific, but it only displays the GOT related contents and it is ignored when --arch-specific is used.

-I

--histogram

Display a histogram of bucket list lengths when displaying the contents of the symbol tables.

-v

--version

Display the version number of readelf.

-W

--wide

Don't break output lines to fit into 80 columns. By default readelf breaks section header and segment listing lines for 64-bit ELF files, so that they fit into 80 columns.

This option causes readelf to print each section header resp. each segment one a single line, which is far more readable on terminals wider than 80 columns.

-T

--silent-truncation

Normally when readelf is displaying a symbol name, and it has to truncate the name to fit into an 80 column display, it will add a suffix of "[...]" to the name. This command line option disables this behaviour, allowing 5 more characters of the name to be displayed and restoring the old behaviour of readelf (prior to release 2.35).

-H

--help

Display the command-line options understood by readelf.

@file

Read command-line options from file. The options read are inserted in place of the original @file option. If file does not exist, or cannot be read, then the option will be treated literally, and not removed.

Options in file are separated by whitespace. A whitespace character may be included in an option by surrounding the entire option in either single or double quotes. Any character (including a backslash) may be included by prefixing the character to be included with a backslash. The file may itself contain additional @file options; any

such options will be processed recursively.

SEE ALSO

objdump(1), and the Info entries for binutils.

COPYRIGHT