



Navigation

• [LLVM Home](#) • | [Documentation](#) • » [Reference](#) • » [LLVM Command Guide](#) » [previous](#) • | [next](#) • | [index](#)

llvm-strip – object stripping tool ¶

SYNOPSIS ¶

llvm-strip [options] inputs...

DESCRIPTION ¶

llvm-strip is a tool to strip sections and symbols from object files. If no other stripping or remove options are specified, [--strip-all](#) will be enabled.

By default, the input files are modified in-place. If “-” is specified for the input file, the input is read from the program’s standard input stream.

If the input is an archive, any requested operations will be applied to each archive member individually.

The tool is still in active development, but in most scenarios it works as a drop-in replacement for GNU’s strip.

GENERIC AND CROSS-PLATFORM OPTIONS ¶

The following options are either agnostic of the file format, or apply to multiple file formats.

`--disable-deterministic-archives, -U ¶`

Use real values for UIDs, GIDs and timestamps when updating archive member headers.

`--discard-all, -x ¶`

Remove most local symbols from the output. Different file formats may limit this to a subset of the local symbols. For example, file and section symbols in ELF objects will not be discarded.

`--enable-deterministic-archives, -D ¶`

Enable deterministic mode when stripping archives, i.e. use 0 for archive member header UIDs, GIDs and timestamp fields. On by default.

`--help, -h ¶`

Print a summary of command line options.

`--no-strip-all ¶`

Disable [--strip-all](#).

Documentation

- [Getting Started/Tutorials](#)
- [User Guides](#)
- [Reference](#)

Getting Involved

- [Contributing to LLVM](#)
- [Submitting Bug Reports](#)
- [Mailing Lists](#)
- [IRC](#)
- [Meetups and Social Events](#)

Additional Links

- [FAQ](#)
- [Glossary](#)
- [Publications](#)
- [Github Repository](#)

This Page

- [Show Source](#)

Quick search

`-o <file>` ¶

Write output to `<file>`. Multiple input files cannot be used in combination with `-o`.

`--regex` ¶

If specified, symbol and section names specified by other switches are treated as extended POSIX regular expression patterns.

`--remove-section <section>`, `-R` ¶

Remove the specified section from the output. Can be specified multiple times to remove multiple sections simultaneously.

`--strip-all-gnu` ¶

Remove all symbols, debug sections and relocations from the output. This option is equivalent to GNU strip's `--strip-all` switch.

`--strip-all`, `-S` ¶

For ELF objects, remove from the output all symbols and non-alloc sections not within segments, except for `.gnu.warning`, `.ARM.attribute` sections and the section name table.

For COFF objects, remove all symbols, debug sections, and relocations from the output.

`--strip-debug`, `-g` ¶

Remove all debug sections from the output.

`--strip-symbol <symbol>`, `-N` ¶

Remove all symbols named `<symbol>` from the output. Can be specified multiple times to remove multiple symbols.

`--strip-unnneeded` ¶

Remove from the output all local or undefined symbols that are not required by relocations. Also remove all debug sections.

`--version`, `-V` ¶

Display the version of the `llvm-strip` executable.

`@<FILE>` ¶

Read command-line options and commands from response file `<FILE>`.

`--wildcard`, `-w` ¶

Allow wildcard syntax for symbol-related flags. On by default for section-related flags. Incompatible with `-regex`.

Wildcard syntax allows the following special symbols:

Character	Meaning	Equivalent
-----------	---------	------------

Character	Meaning	Equivalent
*	Any number of characters	.*
?	Any single character	.
\	Escape the next character	\
[a-z]	Character class	[a-z]
[!a-z], [^a-z]	Negated character class	[^a-z]

Additionally, starting a wildcard with '!' will prevent a match, even if another flag matches. For example `-w -N '*' -N '!x'` will strip all symbols except for `x`.

The order of wildcards does not matter. For example, `-w -N '*' -N '!x'` is the same as `-w -N '!x' -N '*'`.

COFF-SPECIFIC OPTIONS ¶

The following options are implemented only for COFF objects. If used with other objects, `llvm-strip` will either emit an error or silently ignore them.

`--only-keep-debug` ¶

Remove the contents of non-debug sections from the output, but keep the section headers.

ELF-SPECIFIC OPTIONS ¶

The following options are implemented only for ELF objects. If used with other objects, `llvm-strip` will either emit an error or silently ignore them.

`--allow-broken-links` ¶

Allow `llvm-strip` to remove sections even if it would leave invalid section references. Any invalid `sh_link` fields will be set to zero.

`--discard-locals, -X` ¶

Remove local symbols starting with ".L" from the output.

`--keep-file-symbols` ¶

Keep symbols of type `STT_FILE`, even if they would otherwise be stripped.

`--keep-section <section>` ¶

When removing sections from the output, do not remove sections named `<section>`. Can be specified multiple times to keep multiple sections.

`--keep-symbol <symbol>, -K` ¶

When removing symbols from the output, do not remove symbols named `<symbol>`. Can be specified multiple times to keep multiple symbols.

`--preserve-dates, -p` ¶

Preserve access and modification timestamps in the output.

`--strip-sections` ¶

Remove from the output all section headers and all section data not within segments. Note that many tools will not be able to use an object without section headers.

EXIT STATUS ¶

`llvm-strip` exits with a non-zero exit code if there is an error. Otherwise, it exits with code 0.

BUGS ¶

To report bugs, please visit <<http://llvm.org/bugs/>>.

SEE ALSO ¶

[llvm-objcopy\(1\)](#)

Navigation

• [LLVM Home](#) • | [Documentation](#) • » [Reference](#) • » [LLVM Command Guide](#) » [previous](#) • | [next](#) • | [index](#)