

NAME

bison – GNU Project parser generator (yacc replacement)

SYNOPSIS

bison [*OPTION*]... *FILE*

DESCRIPTION

Bison is a parser generator in the style of *yacc*(1). It should be upwardly compatible with input files designed for *yacc*.

Input files should follow the *yacc* convention of ending in **.y**. Unlike *yacc*, the generated files do not have fixed names, but instead use the prefix of the input file. Moreover, if you need to put C++ code in the input file, you can end his name by a C++-like extension (**.ypp** or **.y++**), then bison will follow your extension to name the output file (**.cpp** or **.c++**). For instance, a grammar description file named **parse.yxx** would produce the generated parser in a file named **parse.tab.cxx**, instead of *yacc*'s **y.tab.c** or old *Bison* version's **parse.tab.c**.

This description of the options that can be given to *bison* is adapted from the node **Invocation** in the **bison.texti** manual, which should be taken as authoritative.

Bison supports both traditional single-letter options and mnemonic long option names. Long option names are indicated with **--** instead of **-**. Abbreviations for option names are allowed as long as they are unique. When a long option takes an argument, like **--file-prefix**, connect the option name and the argument with **=**.

Generate a deterministic LR or generalized LR (GLR) parser employing LALR(1), IELR(1), or canonical LR(1) parser tables.

Mandatory arguments to long options are mandatory for short options too. The same is true for optional arguments.

Operation Modes:

- h, --help**
display this help and exit
- V, --version**
output version information and exit
- print-localedir**
output directory containing locale-dependent data and exit
- print-datadir**
output directory containing skeletons and XSLT and exit
- u, --update**
apply fixes to the source grammar file and exit
- f, --feature[=FEATURES]**
activate miscellaneous features

FEATURES is a list of comma separated words that can include:

- caret, diagnostics-show-caret
show errors with carets
- fixit, diagnostics-parseable-fixits
show machine-readable fixes
- syntax-only
do not generate any file
- all
all of the above
- none
disable all of the above

Diagnostics:

- W, --warnings[=CATEGORY]**
report the warnings falling in CATEGORY
- color[=WHEN]**
whether to colorize the diagnostics
- style=FILE**
specify the CSS FILE for colorizer diagnostics

Warning categories include:

- conflicts-sr**
S/R conflicts (enabled by default)
- conflicts-rr**
R/R conflicts (enabled by default)
- counterexamples, cex**
generate conflict counterexamples
- dangling-alias**
string aliases not attached to a symbol
- deprecated**
obsolete constructs
- empty-rule**
empty rules without %empty
- midrule-values**
unset or unused midrule values
- precedence**
useless precedence and associativity
- yacc** incompatibilities with POSIX Yacc
- other** all other warnings (enabled by default)
- all** all the warnings except 'counterexamples', 'dangling-alias' and 'yacc'
- no-CATEGORY**
turn off warnings in CATEGORY
- none** turn off all the warnings
- error[=CATEGORY]**
treat warnings as errors

WHEN can be one of the following:

- always, yes**
colorize the output
- never, no**
don't colorize the output
- auto, tty**
colorize if the output device is a tty

Tuning the Parser:

- L, --language=LANGUAGE**
specify the output programming language
- S, --skeleton=FILE**
specify the skeleton to use

- t, --debug**
instrument the parser for tracing same as `'-Dparse.trace'`
- locations**
enable location support
- D, --define=NAME[=VALUE]**
similar to `'%define NAME VALUE'`
- F, --force-define=NAME[=VALUE]**
override `'%define NAME VALUE'`
- p, --name-prefix=PREFIX**
prepend PREFIX to the external symbols deprecated by `'-Dapi.prefix={PREFIX}'`
- l, --no-lines**
don't generate `'#line'` directives
- k, --token-table**
include a table of token names
- y, --yacc**
emulate POSIX Yacc

Output Files:

- H, --header=[FILE]**
also produce a header file
- d** likewise but cannot specify FILE (for POSIX Yacc)
- r, --report=THINGS**
also produce details on the automaton
- report-file=FILE**
write report to FILE
- v, --verbose**
same as `'--report=state'`
- b, --file-prefix=PREFIX**
specify a PREFIX for output files
- o, --output=FILE**
leave output to FILE
- g, --graph=[FILE]**
also output a graph of the automaton
- html=[FILE]**
also output an HTML report of the automaton
- x, --xml=[FILE]**
also output an XML report of the automaton
- M, --file-prefix-map=OLD=NEW** replace prefix OLD with NEW when writing file paths in output files

THINGS is a list of comma separated words that can include:

- states describe the states
- itemsets
complete the core item sets with their closure
- lookaheads
explicitly associate lookahead tokens to items

solved describe shift/reduce conflicts solving
counterexamples, cex
generate conflict counterexamples
all include all the above information
none disable the report

AUTHOR

Written by Robert Corbett and Richard Stallman.

REPORTING BUGS

Report bugs to <bug-bison@gnu.org>.
GNU Bison home page: <<https://www.gnu.org/software/bison/>>.
General help using GNU software: <<https://www.gnu.org/gethelp/>>.
For complete documentation, run: info bison.

COPYRIGHT

Copyright © 2021 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

SEE ALSO

lex(1), **flex(1)**, **yacc(1)**.

The full documentation for **bison** is maintained as a Texinfo manual. If the **info** and **bison** programs are properly installed at your site, the command

info bison

should give you access to the complete manual.