NAME

valac - compiler that translates Vala source code into C source and header files

SYNOPSIS

valac [*OPTION*]... [*FILE*]...

DESCRIPTION

Vala is a programming language that aims to bring modern programming language features to GNOME developers without imposing any additional runtime requirements and without using a different ABI compared to applications and libraries written in C.

valac, the Vala compiler, is a self-hosting compiler that translates

Vala source code into C source and header files. It uses the GObject type system to create classes and interfaces declared in the Vala source code.

Usage:

valac [OPTION?] FILE... - Vala Compiler

Help Options:

-?, **--help**

Show help options

Application Options:

--vapidir=*DIRECTORY*...

Look for package bindings in DIRECTORY

--girdir=*DIRECTORY*...

Look for .gir files in DIRECTORY

--metadatadir=*DIRECTORY*...

Look for GIR .metadata files in DIRECTORY

--pkg=*PACKAGE*...

Include binding for PACKAGE

--vapi=*FILE*

Output VAPI file name

--library=*NAME*

Library name

--shared-library=*NAME*
Shared library name used in generated gir

--gir=*NAME-VERSION*.gir
GObject-Introspection repository file name

-b, **--basedir**=*DIRECTORY*
Base source directory

-d, **--directory**=*DIRECTORY*
Change output directory from current working directory

--version
Display version number

--api-version
Display API version number

-C, **--ccode**
Output C code

-H, **--header**=*FILE*
Output C header file

--use-header
Use C header file (DEPRECATED AND IGNORED)

--includedir=*DIRECTORY*
Directory used to include the C header file

-h, **--internal-header**=*FILE*
Output internal C header file

--internal-vapi=*FILE*
Output vapi with internal api

--fast-vapi
Output vapi without performing symbol resolution

--use-fast-vapi

Use **--fast-vapi** output during this compile

--vapi-comments
Include comments in generated vapi

--deps
Write make-style dependency information to this file

--depfile
Write make-style external dependency information for build systems to this file

--list-sources
Output a list of all source and binding files which are used

--symbols=*FILE*
Output symbols file

-c, **--compile**
Compile but do not link

-o, **--output**=*FILE*
Place output in file FILE

-g, **--debug**
Produce debug information

--thread
Enable multithreading support (DEPRECATED AND IGNORED)

--enable-mem-profiler
Enable GLib memory profiler

-D, **--define**=*SYMBOL*...
Define SYMBOL

--main=*SYMBOL*...
Use SYMBOL as entry point

--nostdpkg
Do not include standard packages

--disable-assert
Disable assertions

--enable-checking
Enable additional run-time checks

--enable-deprecated
Enable deprecated features

--hide-internal
Hide symbols marked as internal

--enable-experimental
Enable experimental features

--disable-warnings
Disable warnings

--fatal-warnings
Treat warnings as fatal

--disable-since-check
Do not check whether used symbols exist in local packages

-k, **--keep-going**
Continue as much as possible after an error

--enable-experimental-non-null
Enable experimental enhancements for non-null types

--enable-gobject-tracing
Enable GObject creation tracing

--cc=*COMMAND*
Use COMMAND as C compiler command

-X, **--Xcc**=*OPTION*...
Pass OPTION to the C compiler

--pkg-config=*COMMAND*

Use COMMAND as pkg-config command

--dump-tree=*FILE*
Write code tree to FILE

--save-temps
Keep temporary files

--profile=*PROFILE*
Minimum runtime dependency: 'gobject' (default) or 'posix' (minimal libc)
gobject enables GLib's GType runtime type system. The runtime environment will usually require libgobject and its small number of dependencies. *posix* removes the dependency on GLib and disables the runtime type system. The profile either generates alternative code or errors at compile time if a Vala language feature is used that requires the runtime type system. This is useful for writing code, for example, that targets microcontrollers or for extremely small system utilities or container images. The runtime environment will usually require a small subset of the ISO C standard library.

-q, **--quiet**
Do not print messages to the console

-v, **--verbose**
Print additional messages to the console

--no-color
Disable colored output, alias for **--color**=*never*

--color=*WHEN*
Enable color output, options are 'always', 'never', or 'auto'
When no value is given *always* is implied. When neither **--color** or **--no-color** are declared then **--color**=*auto* is used where output is colored when stderr is a terminal.

--target-glib='MAJOR.MINOR', or 'auto'
Target version of glib for code generation

--gresources=*FILE...*
XML of gresources

--gresourcesdir=*DIRECTORY...*
Look for resources in DIRECTORY

--enable-version-header
Write vala build version in generated files

--disable-version-header
Do not write vala build version in generated files

--run-args
Arguments passed to directly compiled executable

--abi-stability
Enable support for ABI stability
This changes the current behaviour to output public members of classes and interfaces the same order as they appear in Vala source. For libraries is recommended to use **--abi-stability** to ensure the maintainability of the resulting Application Binary Interface (ABI). This option is disabled by default for backward compatibility because it can break ABI of existing projects.

BUGS
https://gitlab.gnome.org/GNOME/vala/issues

HOMEPAGE OR CONTACT
https://wiki.gnome.org/Projects/Vala

FEATURES
Interfaces, properties, signals, foreach, lambda expressions, type inference for local variables, generics, non-null types, assisted memory management, exception handling

AUTHORS
J??rg Billeter, Raffaele Sandrini, Rico Tzschichholz.