NAME

g-ir-scanner - Extracting C metadata from sources and headers

SYNOPSIS

g-ir-scanner [OPTION...] FILES...

DESCRIPTION

g-ir-scanner is a tool which generates GIR XML files by parsing headers and introspecting GObject based libraries. It is usually invoked during the normal build step for a project and the information is saved to disk and later installed, so that language bindings and other applications can use it. Header files and source files are passed in as arguments on the command line. The suffix determines whether a file be treated as a source file (.c) or a header file (.h). Currently only C based libraries are supported by the scanner.

OPTIONS

--help

Show help options

--quiet

If passed, do not print details of normal operation.

--warn-all

Display warnings for public API which is not introspectable.

--warn-error

Make warnings be fatal errors.

--strict

Display warnings for strict introspectable API.

--format=_FORMAT_

This parameters decides which the resulting format will be used. The default value is gir.

--include=_NAME_

Add the specified introspection dependency to the scanned namespace. NAME is of the form NAMESPACE-VERSION, like Gtk-3.0.

--include-uninstalled=_PATH_

Add the specified introspection dependency to the scanned namespace. This differs from **--include**

in that it takes a file path, and does not process the pkg-config dependencies (since they may not be installed yet).

--add-include-path=*PATH*

Add a directory to the path which the scanner uses to find GIR files. Can be used multiple times to specify multiple directories

-i,*--library=***LIBRARY**

Specifies a library that will be introspected. This means that the *_get_type() functions in it will be called for GObject data types. The name of the library should not contain the leading lib prefix nor the ending shared library suffix.

-L,*--library-path=***PATH**

Include this directory when searching for a library. This option can be specified multiple times to include more than one directory to look for libraries in.

-I*directory*

Include this directory in the list of directories to be searched for header files. You need to pass to the scanner all the directories you'd normally pass to the compiler when using the specified source files.

--c-include=*C_INCLUDES*

Headers which should be included in C programs. This option can be specified multiple times to include more than one header.

-n,*--namespace=***NAME**

The namespace name. This name should be capitalized, eg the first letter should be upper case. Examples: Gtk, Clutter, WebKit.

--no-libtool

Disable usage of libtool for compiling stub introspection binary. Use this if your build system does not require libtool.

--libtool

Full path to libtool executable. Typically used for Automake systems.

--nsversion=*VERSION*

The namespace version. For instance 1.0. This is usually the platform version, eg 2.0 for Gtk+, not 2.12.7.

-p,*--program*=**PROGRAM**

Specifies a binary that will be introspected. This means that the *_get_type() functions in it will be called for GObject data types. The binary must be modified to take a **--introspect-dump**= option, and to pass the argument to this function to g_irepository_dump.

--program-arg=*ARG*

Additional argument to pass to program for introspection.

--identifier-prefix=*PREFIX*

This option may be specified multiple times. Each one gives a prefix that will be stripped from all C identifiers. If none specified, the namespace will be used. Eg, an identifier prefix of Foo will export the identifier typdef struct _FooBar FooBar; as Foo.Bar.

--symbol-prefix=*PREFIX*

This option may be specified multiple times. Each one gives a prefix that will be stripped from all C symbols. Eg, an symbol prefix of foo will export the symbol foo_bar_do_something as Foo.Bar.do_something.

--accept-unprefixed

If specified, the scanner will accept identifiers and symbols which do not match the namespace prefix. Try to avoid using this if possible.

--output=*FILENAME*

Name of the file to output. Normally namespace + format extension. Eg, GLib-2.0.gir.

--pkg=*PACKAGE*

List of pkg-config packages to get compiler and linker flags from. This option can be specified multiple times to include flags from several pkg-config packages.

--pkg-export=*PACKAGE*

List of pkg-config packages that are provided by the generated gir. This option can be specified multiple times if the gir provides more packages. If not specified, the packages specified with **--pkg**= will be used.

--compiler=*COMPILER*

The C compiler to be used internally by g-ir-scanner when introspecting the run time type information, like properties, signals, ancestors, and implemented interfaces. It has the same semantics as the **CC** environment variable.

--verbose

Be verbose, include some debugging information.

ENVIRONMENT VARIABLES

The g-ir-scanner uses the **XDG_DATA_DIRS** variable to check for dirs, the girs are located in **XDG_DATA_DIRS/gir-1.0**. It is normally set on a distribution so you shouldn't need to set it yourself.

The variable **GI_SCANNER_DISABLE_CACHE** ensures that the scanner will not write cache data to **$HOME**.

The variable **GI_SCANNER_DEBUG** can be used to debug issues in the build-system that involve g-ir-scanner. When it is set to **save-temps**, then g-ir-scanner will not remove temporary files and directories after it terminates.

The variable **GI_HOST_OS** can be used to control the OS name on the host that runs the scanner. It has the same semantics as the Python **os.name** property.

The variable **GI_CROSS_LAUNCHER** can be used to wrap the GType introspection binary generated by g-ir-scanner before executing it. It is useful when generating introspection data in a cross-compilation environment.

BUGS

Report bugs at *https://gitlab.gnome.org/GNOME/gobject-introspection/issues*

HOMEPAGE AND CONTACT

https://gi.readthedocs.io/

AUTHORS

Johan Dahlin