

query format

In this article

- Synopsis
- Description
- Options


Autoformat QL source code.

GitHub CodeQL is licensed on a per-user basis upon installation. You can use CodeQL only for certain tasks under the license restrictions. For more information, see "[About the CodeQL CLI](#)." If you have a GitHub Advanced Security license, you can use CodeQL for automated analysis, continuous integration, and continuous delivery. For more information, see "[About GitHub Advanced Security](#)."

This content describes the most recent release of the CodeQL CLI. For more information about this release, see <https://github.com/github/codeql-cli-binaries/releases>.

To see details of the options available for this command in an earlier release, run the command with the `--help` option in your terminal.

Synopsis

Shell 


```
codeql query format [--output=<file>] [--in-place] [--backup=<ext>] <options>...
-- <file>...
```

Description


Autoformat QL source code.

Options

Primary Options

<file>... 

One or more `.ql` or `.qll` source files to autoformat. A dash can be specified to read from standard input.

-o, --output=<file> 

Write the formatted QL code to this file instead of the standard output stream. Must not

be given if there is more than one input.

-i, --[no-]in-place [↗](#)

Overwrite each input file with a formatted version of its content.

--[no-]check-only [↗](#)

Instead of writing output, exit with status 1 if any input files *differ* from their correct formatting. A message telling which files differed will be printed to standard error unless you also give **-qq**.

-b, --backup=<ext> [↗](#)

When writing a file that already exists, rename the existing file to a backup by appending this extension to its name. If the backup file already exists, it will be silently deleted.

--no-syntax-errors [↗](#)

If an input file is not syntactically correct QL, pretend that it is already correctly formatted. (Usually such a file causes the command to terminate with an error message).

Common options [↗](#)

-h, --help [↗](#)

Show this help text.

-J=<opt> [↗](#)

[Advanced] Give option to the JVM running the command.

(Beware that options containing spaces will not be handled correctly.)

-v, --verbose [↗](#)

Incrementally increase the number of progress messages printed.

-q, --quiet [↗](#)

Incrementally decrease the number of progress messages printed.

--verbosity=<level> [↗](#)

[Advanced] Explicitly set the verbosity level to one of errors, warnings, progress, progress+, progress++, progress+++. Overrides **-v** and **-q**.

--logdir=<dir> [↗](#)

[Advanced] Write detailed logs to one or more files in the given directory, with generated names that include timestamps and the name of the running subcommand.

(To write a log file with a name you have full control over, instead give **--log-to-stderr** and redirect stderr as desired.)

Legal

