# resolve library-path

**In this article**



[Deep plumbing] Determine QL library path and dbscheme for a query.



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 resolve library-path (--query=<qlfile> | --dir=<dir> | --root-pack=
<pkgname>) <options>...
```

## Description

[Deep plumbing] Determine QL library path and dbscheme for a query.

Determine which QL library path a particular query should be compiled against. This computation is implicit in several subcommands that may need to compile queries. It is exposed as a separate plumbing command in order to (a) help with troubleshooting, and (b) provide a starting point for modifying the path in extraordinary cases where exact control is needed.

The command will also detect a language and dbscheme to compile a query against, as these may also depend on autodetecting the language of a QL query.

**The command is deeply internal and its behavior or existence may change without much notice as the QL language ecosystem evolves.**

## Options

## Primary Options

### `--[no-]find-extractors`

[Advanced] Include in the output a summary of `extractor` fields from the QL packs that the query depends on. This is used only for a few rare internal cases, and may require more work to compute, so is not turned on by default.

### `--format=<fmt>`

Select output format. Choices include:

`lines` *(default)*: Print command line arguments on one line each.

`json`: Print a JSON object with all the data.

`path`: Print just the computed library path.

`dbscheme`: Print just the detected dbscheme.

`cache`: Print the default compilation cache location, or nothing if none.

## Options from the invoking command's command line

### `--search-path=<dir>[:<dir>...]`

A list of directories under which QL packs may be found. Each directory can either be a QL pack (or bundle of packs containing a `.codeqlmanifest.json` file at the root) or the immediate parent of one or more such directories.

If the path contains more than one directory, their order defines precedence between them: when a pack name that must be resolved is matched in more than one of the directory trees, the one given first wins.

Pointing this at a checkout of the open-source CodeQL repository ought to work when querying one of the languages that live there.

If you have checked out the CodeQL repository as a sibling of the unpacked CodeQL toolchain, you don't need to give this option; such sibling directories will always be searched for QL packs that cannot be found otherwise. (If this default does not work, it is strongly recommended to set up `--search-path` once and for all in a per-user configuration file).

(Note: On Windows the path separator is `;`).

### `--additional-packs=<dir>[:<dir>...]`

If this list of directories is given, they will be searched for packs before the ones in `--search-path`. The order between these doesn't matter; it is an error if a pack name is found in two different places through this list.

This is useful if you're temporarily developing a new version of a pack that also appears in the default path. On the other hand, it is *not recommended* to override this option in a config file; some internal actions will add this option on the fly, overriding any configured value.

(Note: On Windows the path separator is `;`).

### `--library-path=<dir>[:<dir>...]`

[Advanced] An optional list of directories that will be added to the raw import search path for QL libraries. This should only be used if you're using QL libraries that have not been packaged as QL packs.

(Note: On Windows the path separator is `;`).

#### `--dbscheme=<file>`

[Advanced] Explicitly define which dbscheme queries should be compiled against. This should only be given by callers that are extremely sure what they're doing.

#### `--compilation-cache=<dir>`

[Advanced] Specify an additional directory to use as a compilation cache.

#### `--no-default-compilation-cache`

[Advanced] Don't use compilation caches in standard locations such as in the QL pack containing the query or in the CodeQL toolchain directory.

### Options for configuring the CodeQL package manager

#### `--registries-auth-stdin`

Authenticate to GitHub Enterprise Server Container registries by passing a comma-separated list of <registry_url>=<token> pairs.

For example, you can pass `https://containers.GHEHOSTNAME1/v2/=TOKEN1,https://containers.GHEHOSTNAME2/v2/=TOKEN2` to authenticate to two GitHub Enterprise Server instances.

This overrides the CODEQL_REGISTRIES_AUTH and GITHUB_TOKEN environment variables. If you only need to authenticate to the github.com Container registry, you can instead authenticate using the simpler `--github-auth-stdin` option.

#### `--github-auth-stdin`

Authenticate to the github.com Container registry by passing a github.com GitHub Apps token or personal access token via standard input.

To authenticate to GitHub Enterprise Server Container registries, pass `--registries-auth-stdin` or use the CODEQL_REGISTRIES_AUTH environment variable.

This overrides the GITHUB_TOKEN environment variable.

### Options for specifying what we're about to compile

Exactly one of these options must be given.

#### `--query=<qlfile>`

The path to the QL file we want to compile.

Its directory and parent directories will be searched for qlpack.yml or legacy queries.xml files to determine necessary packs.

#### `--dir=<dir>`

The root directory of the pack containing queries to compile.

#### `--root-pack=<pkgname>`

[Advanced] The declared name of a pack to use as root for dependency resolution.

This is used when the pack can be found by name somewhere in the search path. If you know the *disk location* of your desired root package, pretend it contains a .ql file and use `--query` instead.

### 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.)