About CodeQL workspaces

In this article

About CodeQL workspaces
The codeql-workspace.yml file
Source dependencies
CodeQL workspaces and query resolution

CodeQL workspaces allow you to develop and maintain a group of CodeQL packs that depend on each other.

About CodeQL workspaces

You use a CodeQL workspace when you want to group multiple CodeQL packs together. A typical use case for a CodeQL workspace is to develop a set of CodeQL library and query packs that are mutually dependent. For more information on CodeQL packs, see "Customizing analysis with CodeQL packs."

The main benefit of a CodeQL workspace is that it makes it easier for you to develop and maintain multiple CodeQL packs. When you use a CodeQL workspace, all the CodeQL packs in the workspace are available as *source dependencies* for each other when you run a CodeQL command that resolves queries. This makes it easier to develop, maintain, and publish multiple, related CodeQL packs.

In most cases, you should store the CodeQL workspace and the CodeQL packs contained in it in one git repository. This makes it easier to share your CodeQL development environment.

The `codeql-workspace.yml` file

A CodeQL workspace is defined by a `codeql-workspace.yml` yaml file. This file contains a `provide` block, and optionally `ignore` and `registries` blocks.

- The `provide` block contains a list of glob patterns that define the CodeQL packs that are available in the workspace.
- The `ignore` block contains a list of glob patterns that define CodeQL packs that are not available in the workspace.
- The `registries` block contains a list of GHES URLs and package patterns that control which container registry is used for publishing CodeQL packs. For more information, see "Publishing and using CodeQL packs."

Each entry in the `provide` or `ignore` section must map to the location of a `qlpack.yml` file. All glob patterns are defined relative to the directory that contains the workspace file. For a list of patterns accepted in this file, see "@actions/glob ."

For example, the following `codeql-workspace.yml` file defines a workspace that contains all the CodeQL packs recursively found in the `codeql-packs` directory, except for the packs in the `experimental` directory. The `registries` block specifies that `codeql/\*` packs should be downloaded from `https://ghcr.io/v2/`, which is GitHub's default container registry. All other packs should be downloaded from and published to the registry at `GHE_HOSTNAME`.

```
provide:
  - "*/codeql-packs/**/qlpack.yml"
ignore:
  - "*/codeql-packs/**/experimental/**/qlpack.yml"

registries:
 - packages: 'codeql/*'
   url: https://ghcr.io/v2/

 - packages: '*'
   url: https://containers.GHE_HOSTNAME/v2/
```

To verify that your `codeql-workspace.yml` file includes the CodeQL packs that you expect, run the `codeql pack ls` command in the same directory as your workspace. The result of the command is a list of all CodeQL packs in the workspace.

Source dependencies

Source dependencies are CodeQL packs that are resolved from the local file system outside of the CodeQL package cache. These dependencies can be in the same CodeQL workspace, or specified as a path option using the `--additional-packs` argument. When you compile and run queries locally, source dependencies override any dependencies found in the CodeQL package cache as well as version constraints defined in the `qlpack.yml`. All references to CodeQL packs in the same workspace are resolved as source dependencies.

This is particularly useful in the following situations:

- One of the dependencies of the query pack you are running is not yet published. Resolving from source is the only way to reference that pack.
- You are making changes to multiple packs at the same time and want to test them together. Resolving from source ensures that you are using the version of the pack with your changes in it.

CodeQL workspaces and query resolution

All CodeQL packs in a workspace are available as source dependencies for each other when you run any CodeQL command that resolves queries or packs. For example, when you run `codeql pack install` in a pack directory in a workspace, any dependency that can be found in the workspace will be used instead of downloading that dependency to the package cache and adding it to the `codeql-pack.lock.yml` file. For more information, see "Creating and working with CodeQL packs."

Similarly, when you publish a CodeQL query pack to the GitHub container registry using `codeql pack publish` the command will always use the dependencies from the workspace instead of using dependencies found in the local package cache.

This ensures that any local changes you make to a query library in a dependency are

automatically reflected in any query packs you publish from that workspace.

Example

Consider the following `codeql-workspace.yml` file:

```
provide:
  - "**/qlpack.yml"
```

And the following CodeQL library pack `qlpack.yml` file in the workspace:

```
name: my-company/my-library
library: true
version: 1.0.0
```

And the following CodeQL query pack `qlpack.yml` file in the workspace:

```
name: my-company/my-queries
version: 1.0.0
dependencies:
  my-company/my-library: "*"
  codeql/cpp-all: ~0.2.0
```

Notice that the `dependencies` block for the CodeQL query pack, `my-company/my-queries`, specifies `"*"` as the version of the library pack. Since the library pack is already defined as a source dependency in `codeql-workspace.yml`, the library pack's content is always resolved from inside the workspace. Any version constraint you define will be ignored in this case. We recommend that you use `"*"` for source dependencies to make it clear that the version is inherited from the workspace.

When you execute `codeql pack install` from the query pack directory, an appropriate version of `codeql/cpp-all` is downloaded to the local package cache. Also, a `codeql-pack.lock.yml` file is created that contains the resolved version of `codeql/cpp-all`. The lock file won't contain an entry for `my-company/my-library` since it is resolved from source dependencies. The `codeql-pack.lock.yml` file will look something like this:

```
dependencies:
  codeql/cpp-all:
    version: 0.2.2
```

When you execute `codeql pack publish` from the query pack directory, the `codeql/cpp-all` dependency from the package cache and the `my-company/my-library` from the workspace are bundled with `my-company/my-queries` and published to the GitHub container registry.