Flake URL

Flakes are referred using an URL-like syntax that is documented here.

Links to this page

Use a local directory as flake input

> A Flake URL can not only be Git repositories. They can also refer to local paths. If you have two projects `~/code/foo` and `~/code/bar`, and `bar` depends on `foo`, you can use the following `flake.nix` in `bar` to have it refer to the local `foo` project:

Rapid Introduction to Nix

> URL-like syntax used by the `url` attribute

> The `nix build` command takes as argument a value of the form `<flake-url>#<package-name>`. By default, `.` (which is a flake URL) refers to the current flake. Thus, `nix build .#cowsay` will build the `cowsay` package from the current flake under the current system. `nix build` produces a `./result` symlink that points to the Nix store path containing the package:

First steps with Nix

> A registry is simply a mapping of flake alias to Flake URL.

> You are not required to use a registry. Without a registry, getting a package off nixpkgs would instead involve its fully qualified URL:

> [!info] `nix run` `nix run` command will run the specified package from the flake. Here `nixpkgs` is the flake, followed by the letter `#`, which is followed by the package (Derivation) name `cowsay` that is outputted by that flake. See Flake URL for details on the syntax.

> nixpkgs is not the only way to get software packaged by Nix. As you have seen immediately above, you can install programs from *any* flake by specifying its flake URL to the `nix ?` commands. For example, Emanote (which is used to build this very website) can be executed or installed directly off its flake on GitHub: