Overriding

In Nix, you can customize Nix packages in `pkgs` by using the `override` function, which allows you to define custom build parameters and returns a new derivation with the overridden values. Let's take a look at an example:

```nix
pkgs.fcitx5-rime.override { rimeDataPkgs = [ ./rime-data-flypy ]; }
```

In the above example, we override the `rimeDataPkgs` parameter of the `fcitx5-rime` derivation to use a custom package called `rime-data-flypy`. This creates a new derivation where `rimeDataPkgs` is overridden, while other parameters remain unchanged.

To find out which parameters of a specific package can be overridden, there are a couple of approaches you can follow:

1. Check the source code of the package in the Nixpkgs repository on GitHub, such as `fcitx5-rime.nix`. Make sure to select the appropriate branch, such as `nixos-unstable`, if you are using that branch.
2. Use the `nix repl -f '<nixpkgs>'` command to open a Nix REPL and then enter `:e pkgs.fcitx5-rime`. This opens the source code of the package in your default editor, where you can see all the parameters of the package. To learn the basic usage of `nix repl`, you can type `:?` to see the help information.

By using these methods, you can discover the input parameters of a package and determine which ones can be modified using `override`.

For example, let's take a look at the source code of pkgs.hello:

```nix
{ callPackage
, lib
, stdenv
, fetchurl
, nixos
, testers
, hello
}:

stdenv.mkDerivation (finalAttrs: {

```

```
    pname = "hello";
    version = "2.12.1";

    src = fetchurl {
      url = "mirror://gnu/hello/hello-${finalAttrs.version}.tar.gz";
      sha256 = "sha256-jZkUKv2SV28wsM18tCqNxoCZmLxdYH2Idh9RLibH2yA=";
    };

    doCheck = true;

    # ...
  })
```

In this example, the attributes `pname`, `version`, `src`, and `doCheck` can all be overridden using `overrideAttrs`. For instance:

nix

```
helloWithDebug = pkgs.hello.overrideAttrs (finalAttrs: previousAttrs: {
  doCheck = false;
});
```

In the above code, we use `overrideAttrs` to override the `doCheck` attribute, while leaving other attributes unchanged.

You can also override some default attributes defined in `stdenv.mkDerivation` using `overrideAttrs`. For example:

nix

```
helloWithDebug = pkgs.hello.overrideAttrs (finalAttrs: previousAttrs: {
  separateDebugInfo = true;
});
```

In this case, we override the `separateDebugInfo` attribute, which is defined in `stdenv.mkDerivation`, rather than in the source code of `hello`.

To see all the attributes defined in `stdenv.mkDerivation`, you can check its source code by using `nix repl -f '<nixpkgs>'` and entering `:e stdenv.mkDerivation`.

This will open the source code in your default editor. If you're new to using `nix repl`, you can type `:?` to see the help information.

Loading comments...