Flake Outputs

In `flake.nix`, the `outputs` section defines the different outputs that a flake can produce during its build process. A flake can have multiple outputs simultaneously, which can include but are not limited to the following:

- Nix packages: These are named `apps.<system>.<name>`, `packages.<system>.<name>`, or `legacyPackages.<system>.<name>`. You can build a specific package using the command `nix build .#<name>`.
- Nix helper functions: These are named `lib.<name>` and serve as libraries for other flakes to use.
- Nix development environments: These are named `devShells` and provide isolated development environments. They can be accessed using the command `nix develop`.
- NixOS configurations: These are named `nixosConfiguration` and represent specific NixOS system configurations. You can activate a configuration using the command `nixos-rebuild switch --flake .#<name>`.
- Nix templates: These are named `templates` and can be used as a starting point for creating new projects. You can generate a project using the command `nix flake init --template <reference>`.
- Other user-defined outputs: These outputs can be defined by the user and may be used by other Nix-related tools.

Here's an example excerpt from the NixOS Wiki that demonstrates the structure of the `outputs` section:

```nix
{
  inputs = {
    # ......
  };

  outputs = { self, ... }@inputs: {
    # Executed by `nix flake check`
    checks."<system>"."<name>" = derivation;
    # Executed by `nix build .#<name>`
    packages."<system>"."<name>" = derivation;
    # Executed by `nix build .`
    packages."<system>".default = derivation;
    # Executed by `nix run .#<name>`

    apps."<system>"."<name>" = {
      type = "app";
      program = "<store-path>";
    };
    # Executed by `nix run . -- <args?>`
    apps."<system>".default = { type = "app"; program = "..."; };

    # Formatter (alejandra, nixfmt or nixpkgs-fmt)
    formatter."<system>" = derivation;
    # Used for nixpkgs packages, also accessible via `nix build .#<name>`
    legacyPackages."<system>"."<name>" = derivation;
    # Overlay, consumed by other flakes
    overlays."<name>" = final: prev: { };
    # Default overlay
    overlays.default = {};
    # Nixos module, consumed by other flakes
    nixosModules."<name>" = { config }: { options = {}; config = {}; };
    # Default module
    nixosModules.default = {};
    # Used with `nixos-rebuild --flake .#<hostname>`
    # nixosConfigurations."<hostname>".config.system.build.toplevel must be a de
    nixosConfigurations."<hostname>" = {};
    # Used by `nix develop .#<name>`
    devShells."<system>"."<name>" = derivation;
    # Used by `nix develop`
    devShells."<system>".default = derivation;
    # Hydra build jobs
    hydraJobs."<attr>"."<system>" = derivation;
    # Used by `nix flake init -t <flake>#<name>`
    templates."<name>" = {
      path = "<store-path>";
      description = "template description goes here?";
    };
    # Used by `nix flake init -t <flake>`
    templates.default = { path = "<store-path>"; description = ""; };
  };
}
```

Loading comments...