

Reproducible builds and deployments.

Nix

Powerful package manager for Linux and other Unix systems that makes package management reliable and reproducible. Share your development and build environments across different machines.

NixOS

Linux distribution with a unique approach to package and configuration management. Built on top of the Nix package manager, it is completely declarative, makes upgrading systems reliable, and has many other advantages.

[Get started](#)[Download](#)

```
$ bat default.nix
File: default.nix
1 { pkgs ? import <nixpkgs> {} # here we import the nixpkgs package s
et
2 };
3 pkgs.mkShell { # mkShell is a helper function
4   name="dev-environment"; # this requires a name
5   buildInputs = [ # and a list of packages
6     pkgs.nodejs
7   ];
8   shellHook = '' # bash to run when you enter the shell
9     echo "Start developing..."
10    '';
11 }

$ # Pause the video to understand the default.nix
$ # To enter dev-environment simply run:
$ nix-shell
Start developing...
[nix-shell]$ node -e "console.log(1+1)"
2
[nix-shell]$ # Now go ahead commit default.
```

Current versions

[Nix 2.3.7.](#)[NixOS 20.03.](#)

Reproducible builds and deployments.

Nix

Powerful package manager for Linux and other Unix systems that makes package management reliable and reproducible. Share your development and build environments across different machines.

NixOS

Linux distribution with a unique approach to package and configuration management. Built on top of the Nix package manager, it is completely declarative, makes upgrading systems reliable, and has many other advantages.

[Get started](#)[Download](#)

```
$ bat default.nix
File: default.nix
1 { pkgs ? import <nixpkgs> {} # here we import the nixpkgs package s
et
2 };
3 pkgs.mkShell { # mkShell is a helper function
4   name="dev-environment"; # this requires a name
5   buildInputs = [ # and a list of packages
6     pkgs.nodejs
7   ];
8   shellHook = '' # bash to run when you enter the shell
9     echo "Start developing..."
10    '';
11 }

$ # Pause the video to understand the default.nix
$ # To enter dev-environment simply run:
$ nix-shell
Start developing...
[nix-shell]$ node -e "console.log(1+1)"
2
[nix-shell]$ # Now go ahead commit default.
```

Current versions

[Nix 2.3.7.](#)[NixOS 20.03.](#)

Reproducible builds and deployments.

Nix

Powerful package manager for Linux and other Unix systems that makes package management reliable and reproducible. Share your development and build environments across different machines.

NixOS

Linux distribution with a unique approach to package and configuration management. Built on top of the Nix package manager, it is completely declarative, makes upgrading systems reliable, and has many other advantages.

[Get started](#)[Download](#)

```
$ bat default.nix
File: default.nix
1 { pkgs ? import <nixpkgs> {} # here we import the nixpkgs package s
et
2 };
3 pkgs.mkShell { # mkShell is a helper function
4   name="dev-environment"; # this requires a name
5   buildInputs = [ # and a list of packages
6     pkgs.nodejs
7   ];
8   shellHook = '' # bash to run when you enter the shell
9     echo "Start developing..."
10    '';
11 }

$ # Pause the video to understand the default.nix
$ # To enter dev-environment simply run:
$ nix-shell
Start developing...
[nix-shell]$ node -e "console.log(1+1)"
2
[nix-shell]$ # Now go ahead commit default.
```

Current versions

[Nix 2.3.7.](#)[NixOS 20.03.](#)

Reproducible builds and deployments.

Nix

Powerful package manager for Linux and other Unix systems that makes package management reliable and reproducible. Share your development and build environments across different machines.
xxxxxxxxxxxxxxxx

[Get started](#)[Download](#)

NixOS

Linux distribution with a unique approach to package and configuration management. Built on top of the Nix package manager, it is completely declarative, makes upgrading systems reliable, and has many other advantages.

```
$ bat default.nix
File: default.nix
1 { pkgs ? import <nixpkgs> {} # here we import the nixpkgs package s
et
2 };
3 pkgs.mkShell { # mkShell is a helper function
4   name="dev-environment"; # this requires a name
5   buildInputs = [ # and a list of packages
6     pkgs.nodejs
7   ];
8   shellHook = '' # bash to run when you enter the shell
9     echo "Start developing..."
10    '';
11 }

$ # Pause the video to understand the default.nix
$ # To enter dev-environment simply run:
$ nix-shell
Start developing...
[nix-shell]$ node -e "console.log(1+1)"
2
[nix-shell]$ # Now go ahead commit default.
```

Current versions

[Nix 2.3.7.](#)[NixOS 20.03.](#)