Features

1

Stop looking for what is missing. Let Nix remind you.

Makes sure nothing gets left behind and reduces the risk of incomplete dependencies. Stop worrying if all dependencies are in order. Let Nix remind you about the missing pieces.

Complete dependencies are ...

<explain complete dependencies>

2

Patches? Custom changes? The sky is the limit.

Completely customisable build tool. Get the power to adjust any part of the build environment. All changes are transparent and Nix keeps it that way.

<transparent source binary model>

3

Multiple development environments. No conflicts with the operating system.

Securely install software with no risk of breaking anything down. Nix prevents conflicts not only between multiple environments, but also with the operating system. All packages remain installed side-by-side even if different users install packages with the conflicting dependencies.

<explain /nix/store>

4

Leave nothing behind. Instantly upgrade.

To eliminate the risk of leaving any parts behind, all operations are atomic. Nix sets everything in place and once ready, there are only two possible outcomes. Upgrade was successful or upgrade did not happen.

<link farm>

5

Is the previous version better? Rollback anytime.

Packages are never overwritten, and old versions are always stored. Roll back to any previous version within milliseconds. Yes, again with the trustworthy atomic operation.

<profiles, generations, garbage collection>

6

Be carefree. Develop with no limits.

Gives you the power to build and deploy software consistently on different machines. Save time, be carefree and rollback anytime to the previous version. Nix will make sure all dependencies are in place.

<reproducibility>

7

Simplify your work. Use Nix.

Keep it simple and develop with a language agnostic tool/package manager. Focus on the work and do not lose time with multiple tools/package managers.

<Language agnostic>

8

No surprises. Trust in your deployments.

Surprises only belong to birthdays, not deployments. Explain, why we trust in deployments, why are there no surprises?
Boost efficiency with declarative configuration. Shorten development cycles and solve problems faster. All that in a truly all-inclusive build environment: NixOS.

<declarative configuration / NixOS>
Reproduce, bugs