Devbox Nix dev environments

CONCEPTS

devbox.json lists the packages a project needs; devbox.lock pins exact versions. Commit both so everyone gets the same tools.

devbox.json packages, env, scripts
↓ `devbox add`
devbox.lock pinned versions
↓ `devbox shell`
Nix store installed packages
↓ `PATH`
Shell isolated tools

PROJECT SETUP

`devbox init`
Create devbox.json in this directory

`devbox add <pkg>`
Add a package to the project

`devbox add <pkg>@<version>`
Add a specific version

`devbox rm <pkg>`
Remove a package

`devbox search <pkg>`
List matching packages and versions

`devbox update`
Update all packages within their version tags

`devbox update <pkg>`
Update only the named packages

SHELL AND RUN

`devbox shell`
Start a shell with your packages

`devbox shell --pure`
Isolated shell, almost no inherited env

`devbox run <script>`
Run a script from devbox.json

`devbox run <cmd>`
Run any command inside the environment

SHELL AND RUN (cont.)

`exit`
Leave the devbox shell

DEVBOX.JSON

Packages (list form)

```
{
  "packages": ["go@latest", "nodejs@20"]
}
```

Environment variables

```
"env": {
  "APP_ENV": "dev",
  "BIN": "$PWD/bin"
}
```

Init hook

```
"shell": {
  "init_hook": ["echo Welcome", "export
FOO=bar"]
}
```

Scripts

```
"shell": {
  "scripts": {
    "test": "go test ./...",
    "ci": ["go vet ./...", "go
test ./..."]
  }
}
```

SERVICES

`devbox services ls`
List available services

`devbox services up`
Start all services with process-compose

`devbox services up -b`
Run services in the background

`devbox services stop <svc>`
Stop a service (all if none named)

GENERATE AND GLOBAL

`devbox generate direnv`
Write an .envrc that loads the environment

`devbox generate dockerfile`
Dockerfile that replicates devbox shell