pnpm Fast Node package manager

Registry npm packages → `fetch once` → **Global store** shared files → `hard links` → **node_modules/.pnpm** virtual store → `symlinks` → **node_modules** direct deps only

CONCEPTS

pnpm keeps one content-addressable store on disk and hard-links files into node_modules, so repeat installs are fast and cheap.

node_modules holds only your direct dependencies, so code that imports an undeclared package fails instead of working by accident.

INSTALL AND ADD

pnpm install
Install all dependencies

pnpm add *<pkg>*
Add a dependency

pnpm add -D *<pkg>*
Add a dev dependency

pnpm add *<pkg>*@*<version>*
Add a specific version or tag

pnpm add -g *<pkg>*
Install a global package

pnpm add -E *<pkg>*
Save the exact version

pnpm install --frozen-lockfile
Fail if lockfile needs changes

INSTALL AND ADD (cont.)

pnpm install --prod
Skip devDependencies

REMOVE AND UPDATE

pnpm remove *<pkg>*
Remove a dependency

pnpm update
Update within package.json ranges

pnpm update *<pkg>*
Update one package

pnpm update --latest
Ignore ranges, go to latest

pnpm update -i
Pick updates interactively

pnpm update -i --latest
Pick latest versions interactively

pnpm outdated
List outdated packages

RUN SCRIPTS

pnpm run *<script>*
Run a package script

pnpm *<script>*
Shorthand for pnpm run

pnpm test
Run the test script

RUN SCRIPTS (cont.)

pnpm run *<script>* **--** *<args>*
Pass arguments to a script

pnpm exec *<cmd>*
Run a binary from node_modules/.bin

pnpm dlx *<pkg>* **[args]**
Fetch and run without installing

pnpm create *<starter>*
Scaffold from a create-* starter

pnpm init
Create a package.json

INSPECT

pnpm list
Show installed packages as a tree

pnpm ls --depth *<n>*
Limit tree depth

pnpm ls *<pkg>*
Find where a package is used

pnpm why *<pkg>*
Show what depends on a package

pnpm audit
Check for known vulnerabilities

WORKSPACES

pnpm -r *<cmd>*
Run a command in every package

pnpm --filter *<pkg>* *<cmd>*
Run in one package

pnpm --filter *<pkg>*... *<cmd>*
Package and its dependencies

pnpm --filter ...*<pkg>* *<cmd>*
Package and its dependents

pnpm --filter "...[origin/main]" *<cmd>*
Packages changed since a ref

pnpm -w add -D *<pkg>*
Add to the workspace root

pnpm add *<pkg>* **--workspace**
Link a local workspace package

pnpm -r run *<script>*
Run in dependency order

pnpm-workspace.yaml

```
packages:
  - 'packages/*'
  - '!**/test/**'
```

WORKSPACES (cont.)

Depend on a sibling with "<pkg>": "workspace:*"; pnpm swaps it for a real version on publish.

CATALOGS AND SETTINGS

Shared versions in pnpm-workspace.yaml

```
catalog:
  react: ^19.0.0
catalogs:
  legacy:
    react: ^17.0.2
```

Use them in package.json

```
"react": "catalog:"
"react": "catalog:legacy"
```

pnpm approve-builds
Allow dependency build scripts

STORE AND PATCHING

pnpm store path
Show the store location

pnpm store prune
Delete unreferenced packages from the store

pnpm link *<dir>*
Link a local package into this project