npm Node package manager

CONCEPTS

package.json declares what you want, package-lock.json pins what you got. Commit both.

package.json version ranges
↓ `npm install`
package-lock.json exact versions
↓ `npm ci`
node_modules installed tree
↓ `npm run`
Scripts build, test, start

Use npm ci in CI: it installs exactly the lockfile and fails if package.json disagrees.

INSTALL

`npm install`
Install all dependencies

`npm install <pkg>`
Add a dependency

`npm install -D <pkg>`
Add a dev dependency

`npm ci`
Clean install from the lockfile

`npm install <pkg>@<version>`
Install a specific version or tag

`npm install -g <pkg>`
Install globally

`npm uninstall <pkg>`
Remove a dependency

`npm ci --omit=dev`
Install production deps only

SCRIPTS & NPX

`npm run <script>`
Run a package.json script

`npm run`
List available scripts

SCRIPTS & NPX (cont.)

`npm test`
Run the test script

`npm start`
Run the start script

`npm run <script> -- <args>`
Pass arguments to the script

`npx <pkg> [args]`
Run a package binary

`npm init -y`
Create package.json with defaults

INSPECT & UPDATE

`npm ls`
Show the dependency tree

`npm ls <pkg>`
Who depends on this package?

`npm outdated`
Show packages with newer versions

`npm update`
Update within allowed ranges

AUDIT

`npm audit`
List known vulnerabilities

`npm audit fix`
Apply compatible fixes

VERSION & PUBLISH

`npm version patch`
Bump the patch version

`npm pack --dry-run`
Preview the tarball contents

`npm publish`
Publish the package

WORKSPACES

`npm run <script> -w <name>`
Run a script in one workspace