npm Node package manager

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

CONCEPTS

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

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 install -E *<pkg>*
Save an exact version

INSTALL (cont.)

npm ci --omit=dev
Install production deps only

npm dedupe
Flatten duplicate packages

SCRIPTS & NPX

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

npm run
List available scripts

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 exec -- *<cmd>*
Run a command from the local tree

npm init -y
Create package.json with defaults

INSPECT & UPDATE

npm ls
Show the dependency tree

INSPECT & UPDATE (cont.)

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

npm outdated
Show packages with newer versions

npm update
Update within allowed ranges

npm view *<pkg>* **versions**
List published versions

npm view *<pkg>* **version**
Latest published version

npm explain *<pkg>*
Why is this installed?

AUDIT

npm audit
List known vulnerabilities

npm audit fix
Apply compatible fixes

npm audit --audit-level=high
Fail only at high severity

VERSION & PUBLISH

npm version patch
Bump the patch version

VERSION & PUBLISH (cont.)

npm version *<x.y.z>*
Set an exact version

npm pack --dry-run
Preview the tarball contents

npm login
Log in to the registry

npm publish
Publish the package

npm publish --access public
Publish a scoped package publicly

npm publish --dry-run
Check a publish without sending

WORKSPACES

npm install -w *<name>*
Install in one workspace

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

npm run *<script>* **-ws**
Run in all workspaces

CONFIG

npm config list
Show effective settings

CONFIG (cont.)

npm config get *<key>*
Read one setting

npm config set *<key>*=*<value>*
Write a setting

npm help *<term>*
Open the docs for a topic

RECIPES

New project

```
npm init -y
npm install express
npm install -D typescript
npm pkg set
scripts.start="node
index.js"
```

CI install and test

```
npm ci
npm run build --if-present
npm test
npm audit --audit-level=high
```

Upgrade, then release

```
npm outdated
npm update
npm test
npm version minor
npm publish
```

Typical scripts block

```
# package.json
"scripts": {
  "build": "tsc",
  "test": "node --test",
  "start": "node dist/
index.js"
}
```