

# npm Riferimento Rapido

Gestione pacchetti, script, versioning, pubblicazione, workspace

## Installazione

### Installare npm e Node

```
node -v && npm -v # controlla versioni installate
npm install -g npm@latest # aggiorna npm stesso
npm install --no-install # installa Node LTS via nvm
npm use 20 # passa a Node 20
```

### Comandi di Installazione

```
npm install # Installa tutte le dipendenze da package.json
npm install pkg # Aggiunge pacchetto come dipendenza
npm install -D pkg # Aggiunge pacchetto come devDependency
npm install -g pkg # Installa pacchetto globalmente
npm install pkg@2.1.0 # Installa versione specifica
npm ci # Installazione pulita dal lock file (CI/CD)
npm uninstall pkg # Rimuove un pacchetto
```

## Gestione dei Pacchetti

### Gestire i Pacchetti

```
npm ls # elenca i pacchetti installati
npm ls --depth=0 # solo il livello principale
npm outdated # controlla versioni più recenti
npm update # aggiorna nel range semver
npm audit # controlla vulnerabilità
```

### Comandi di Gestione

```
npm ls # Elenca i pacchetti installati ad albero
npm outdated # Mostra pacchetti con versioni più recenti
npm update [pkg] # Aggiorna pacchetti nel range semver
npm audit # Controlla le dipendenze per vulnerabilità
npm audit fix # Corregge automaticamente le dipendenze vulnerabili
npm prune # Rimuove pacchetti non necessari
npm dedupe # Appiattisce l'albero delle dipendenze per ridurre duplicati
```

## Script

### Eseguire Script

```
npm run build # esegue lo script "build"
npm test # scorciatoia per lo script "test"
npm start # scorciatoia per lo script "start"
npm run lint -- --fix # passa argomenti allo script
npm run dev & # esegue in background
```

### Ciclo di Vita degli Script

```
npm test / npm t # Esegue scripts.test
npm start # Esegue scripts.start
npm run <name> # Esegue qualsiasi script personalizzato
pre<name> # Eseguito automaticamente prima di <name>
post<name> # Eseguito automaticamente dopo <name>
npm run # Elenca tutti gli script disponibili
```

## package.json

### Inizializzazione e Campi

```
npm init # configurazione interattiva
npm init -y # accetta tutti i default
npm pkg set name="my-app" # imposta un campo
npm pkg get version # legge un campo
```

## Campi Principali

```
name # Nome del pacchetto (minuscolo, senza spazi)
version # Versione corrente (semver: major.minor.patch)
main # Entry point per CommonJS (require)
module # Entry point per moduli ES (bundler)
type # "module" per ESM, "commonjs" per CJS (default)
scripts # Comandi nominati (build, test, start, ecc.)
dependencies # Dipendenze di produzione
devDependencies # Dipendenze solo per lo sviluppo
engines # Versioni richieste di Node/npm
```

## Versioning

### Comandi di Versione

```
npm version patch # 1.0.0 -> 1.0.1
npm version minor # 1.0.1 -> 1.1.0
npm version major # 1.1.0 -> 2.0.0
npm version 3.2.1 # imposta versione esplicita
npm version prerelease --preid=beta # 1.0.0-beta.0
```

### Range Semver

```
^1.2.3 # Compatibile: >=1.2.3 <2.0.0 (default)
~1.2.3 # Patch-level: >=1.2.3 <1.3.0
1.2.3 # Solo versione esatta
>=1.0.0 <2.0.0 # Range esplicito
* # Qualsiasi versione
1.x / 1.2.x # Range con wildcard
latest # Tag dell'ultima versione pubblicata
```

## Pubblicazione

### Flusso di Pubblicazione

```
npm login # autenticati nel registry
npm publish # pubblica pacchetto pubblico
npm publish --access public # pacchetto con scope pubblico
npm unpublish pkg@1.0.0 # rimuovi versione specifica
npm deprecate pkg@<2 "Usa v2+" # depreca versioni vecchie
```

### Riferimento alla Pubblicazione

```
npm login # Autenticati con il registry npm
npm publish # Pubblica il pacchetto nel registry
npm pack # Crea tarball senza pubblicare
npm unpublish # Rimuovi versione pubblicata (entro 72h)
npm deprecate # Segna versioni come deprecate
.npmignore # File da escludere dal pacchetto pubblicato
files (package.json) # Allowlist dei file da includere nel pacchetto
```

## Workspace

### Comandi Workspace

```
npm init -w packages/core # crea workspace
npm install -w packages/core lodash # installa nel workspace
npm run build --workspaces # esegui in tutti i workspace
npm run test -w packages/api # esegui in workspace specifico
npm ls --workspaces # elenca dipendenze workspace
```

## Configurazione Workspace

```
workspaces (package.json) # Array di glob workspace: ["packages/*"]
-w / --workspace # Punta a un workspace specifico
--workspaces # Esegui comando su tutti i workspace
--include-workspace-root # Includi il pacchetto root nelle operazioni workspace
npm install (root) # Installa tutte le dipendenze dei workspace
Hoisting # Dipendenze condivise elevate al node_modules root
```

## npx

### Eseguire con npx

```
npx create-react-app my-app # esegui senza installare
npx tsc --init # esegui bin locale o remoto
npx -p typescript tsc file.ts # specifica pacchetto esplicitamente
npx --yes create-next-app # salta prompt installazione
npx node@18 -e "console.log('hi')" # esegui con Node specifico
```

### Opzioni npx

```
npx cmd # Esegui cmd da node_modules/.bin locale o remoto
npx -p pkg cmd # Installa pkg, poi esegui cmd
npx --yes cmd # Conferma automaticamente il prompt di installazione
npx --no cmd # Rifiuta installazione — fallisce se non è locale
npx -c 'cmd' # Esegui comando shell con PATH npx
npx node@ver # Esegui versione specifica di Node.js
```

## Configurazione

### Comandi di Configurazione

```
npm config list # mostra configurazione corrente
npm config set registry https://r.npmjs.com/
npm config set init-author-name "Nome"
npm config get prefix # percorso installazione globale
npm config delete key # rimuovi un valore di configurazione
```

### Riferimento Configurazione

```
.npmrc (project) # File di configurazione per progetto
~/ .npmrc # File di configurazione per utente
registry # URL del registry dei pacchetti
save-exact # true per bloccare versioni esatte all'installazione
engine-strict # true per applicare il campo engines
fund # false per sopprimere i messaggi di funding
audit # false per saltare l'audit all'installazione
```

## Pattern Comuni

### One-Liner

```
npm ls --depth=0 --json | jq '.dependencies | keys[]'
npm outdated --long # mostra tipo e homepage
npm cache clean --force # svuota cache npm
npm explain pkg # perché è installato pkg?
npm exec -- envinfo --system # info sistema per bug report
```

# npm Riferimento Rapido

---

## Ricette

<b>Solo lock file</b>	<code>npm ci</code> — installazione pulita da package-lock.json
<b>Controlla licenze</b>	<code>npx license-checker --summary</code>
<b>Trova dep inutilizzate</b>	<code>npx depcheck</code>
<b>Dimensione bundle</b>	<code>npx bundlephobia-cli pkg</code> — controlla dimensione pacchetto
<b>Aggiorna tutto</b>	<code>npx npm-check-updates -u &amp;&amp; npm install</code>
<b>Registry locale</b>	<code>npx verdaccio</code> — esegui registry privato