

Référence rapide npm

Gestion de paquets, scripts, versionnage, publication, espaces de travail

Installation

Installer npm et Node

```
node -v && npm -v # vérifier les versions
installées
npm install -g npm@latest # mettre à jour npm
npm install --lts # installer Node LTS via npm
npm use 20 # passer à Node 20
```

Commandes d'installation

```
npm install # Installer toutes les dépendances du package.json
npm install pkg # Ajouter un paquet comme dépendance
npm install -D pkg # Ajouter un paquet comme devDependency
npm install -g pkg # Installer un paquet globalement
npm install pkg@2.1.0 # Installer une version spécifique
npm ci # Installation propre depuis le fichier de verrouillage (CI/CD)
npm uninstall pkg # Supprimer un paquet
```

Gestion des paquets

Gérer les paquets

```
npm ls # lister les paquets installés
npm ls --depth=0 # niveau supérieur uniquement
npm outdated # vérifier les nouvelles versions
npm update # mettre à jour dans la plage semver
npm audit # vérifier les vulnérabilités
```

Commandes de gestion

```
npm ls # Lister les paquets installés en arbre
npm outdated # Afficher les paquets avec de nouvelles versions
npm update [pkg] # Mettre à jour dans la plage semver
npm audit # Auditer les dépendances pour vulnérabilités
npm audit fix # Corriger automatiquement les dépendances vulnérables
npm prune # Supprimer les paquets superflus
npm dedupe # Aplatir l'arbre de dépendances
```

Scripts

Exécuter des scripts

```
npm run build # exécuter le script "build"
npm test # raccourci pour le script "test"
npm start # raccourci pour le script "start"
npm run lint -- --fix # passer des arguments au script
npm run dev & # exécuter en arrière-plan
```

Cycle de vie des scripts

```
npm test / npm t # Exécuter scripts.test
npm start # Exécuter scripts.start
npm run <name> # Exécuter tout script personnalisé
pre<name> # S'exécute automatiquement avant <name>
post<name> # S'exécute automatiquement après <name>
npm run # Lister tous les scripts disponibles
```

package.json

Initialiser et champs

```
npm init # configuration interactive
npm init -y # accepter tous les défauts
npm pkg set name="my-app" # définir un champ
npm pkg get version # lire un champ
```

Champs clés

```
name # Nom du paquet (minuscules, sans espaces)
version # Version courante (semver : majeur.mineur.correctif)
main # Point d'entrée pour CommonJS (require)
module # Point d'entrée pour les modules ES (bundlers)
type # "module" pour ESM, "commonjs" pour CJS (défaut)
scripts # Commandes nommées (build, test, start, etc.)
dependencies # Dépendances de production
devDependencies # Dépendances de développement uniquement
engines # Plages de versions Node/npm requises
```

Versionnage

Commandes de version

```
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 # définir une version explicite
npm version prerelease --preid=beta # 1.0.0-beta.0
```

Plages semver

```
^1.2.3 # Compatible : >=1.2.3 <2.0.0 (défaut)
~1.2.3 # Niveau correctif : >=1.2.3 <1.3.0
1.2.3 # Version exacte uniquement
>=1.0.0 <2.0.0 # Plage explicite
* # Toute version
1.x / 1.2.x # Plages avec joker
latest # Dernière étiquette de version publiée
```

Publication

Flux de publication

```
npm login # s'authentifier au registre
npm publish # publier un paquet public
npm publish --access public # paquet scopé comme public
npm unpublish pkg@1.0.0 # supprimer une version spécifique
npm deprecate pkg@"<2" "Use v2+" # déprécier les anciennes versions
```

Référence de publication

```
npm login # S'authentifier avec le registre npm
npm publish # Publier le paquet dans le registre
npm pack # Créer une archive sans publier
npm unpublish # Supprimer une version publiée (dans les 72h)
npm deprecate # Marquer des versions comme dépréciées
.npmignore # Fichiers à exclure du paquet publié
files (package.json) # Liste blanche de fichiers à inclure
```

Espaces de travail

Commandes d'espace de travail

```
npm init -w packages/core # créer un espace de travail
npm install -w packages/core lodash # installer dans l'espace
npm run build --workspaces # exécuter dans tous les espaces
npm run test -w packages/api # exécuter dans un espace spécifique
npm ls --workspaces # lister les dépendances des espaces
```

Configuration des espaces de travail

```
workspaces (package.json) # Tableau de globs d'espaces : ["packages/*"]
-w / --workspace # Cibler un espace de travail spécifique
--workspaces # Exécuter la commande sur tous les espaces
--include-workspace-root # Inclure le paquet racine dans les opérations
npm install (racine) # Installe toutes les dépendances des espaces
Hoisting # Dépendances partagées hissées vers la racine node_modules
```

npmx

Exécuter avec npmx

```
npx create-react-app my-app # exécuter sans installer
npx tsc --init # exécuter un binaire local ou distant
npx -p typescript tsc file.ts # spécifier le paquet explicitement
npx --yes create-next-app # ignorer la confirmation d'installation
npx node@18 -e "console.log('hi')" # exécuter avec un Node spécifique
```

Options npmx

```
npx cmd # Exécuter cmd depuis node_modules/.bin ou à distance
npx -p pkg cmd # Installer pkg, puis exécuter cmd
npx --yes cmd # Confirmer automatiquement l'installation
npx --no cmd # Refuser l'installation — échouer si non local
npx -c 'cmd' # Exécuter une commande shell avec le PATH npmx
npx node@ver # Exécuter une version spécifique de Node.js
```

Configuration

Commandes de configuration

```
npm config list # afficher la configuration actuelle
npm config set registry https://r.npmjs.com/
npm config set init-author-name "Nom"
npm config get prefix # chemin d'installation global
npm config delete key # supprimer une valeur de config
```

Référence de configuration

```
.npmrc (projet) # Fichier de config par projet
~/ .npmrc # Fichier de config par utilisateur
registry # URL du registre de paquets
save-exact # true pour épingler les versions exactes à l'installation
engine-strict # true pour appliquer le champ engines
fund # false pour supprimer les messages de financement
audit # false pour ignorer l'audit à l'installation
```

Motifs courants

Commandes en une ligne

```
npm ls --depth=0 --json | jq '.dependencies | keys[]'
npm outdated --long # afficher le type et la page d'accueil
npm cache clean --force # vider le cache npm
npm explain pkg # pourquoi pkg est installé ?
npm exec -- envinfo --system # infos système pour rapports de bugs
```

Référence rapide npm

Recettes

Fichier de verrouillage uniquement	<code>npm ci</code> — installation propre depuis package-lock.json
Vérifier les licences	<code>npx license-checker --summary</code>
Trouver les dépendances inutilisées	<code>npx depcheck</code>
Taille du bundle	<code>npx bundlephobia-cli pkg</code> — vérifier la taille
Tout mettre à jour	<code>npx npm-check-updates -u && npm install</code>
Registre local	<code>npx verdaccio</code> — exécuter un registre privé