REFERÊNCIA RÁPIDA DE NPM

Gerenciamento de pacotes, scripts, versionamento, publicação, workspaces

Instalação

Instalando npm e Node

```
node -v && npm -v                    # check installed versions
npm install -g npm@latest            # update npm itself
nvm install --lts                    # install Node LTS via nvm
nvm use 20                           # switch to Node 20
```

Comandos de Instalação

`npm install`	Instalar todas as dependências do package.json
`npm install pkg`	Adicionar pacote como dependência
`npm install -D pkg`	Adicionar pacote como devDependency
`npm install -g pkg`	Instalar pacote globalmente
`npm install pkg@2.1.0`	Instalar versão específica
`npm ci`	Instalação limpa a partir do lock file (CI/CD)
`npm uninstall pkg`	Remover pacote

Gerenciamento de Pacotes

Gerenciando Pacotes

```
npm ls                               # list installed packages
npm ls --depth=0                     # top-level only
npm outdated                         # check for newer versions
npm update                           # update within semver range
npm audit                            # check for vulnerabilities
```

Comandos de Gerenciamento

`npm ls`	Listar pacotes instalados em árvore
`npm outdated`	Exibir pacotes com versões mais recentes
`npm update [pkg]`	Atualizar pacotes dentro do intervalo semver
`npm audit`	Auditar dependências em busca de vulnerabilidades
`npm audit fix`	Corrigir dependências vulneráveis automaticamente
`npm prune`	Remover pacotes desnecessários
`npm dedupe`	Achatar árvore de dependências para reduzir duplicação

Scripts

Executando Scripts

```
npm run build                        # run "build" script
npm test                             # shortcut for "test" script
npm start                            # shortcut for "start" script
npm run lint -- --fix                # pass args to script
npm run dev &                        # run in background
```

Ciclo de Vida de Scripts

`npm test / npm t`	Executar `scripts.test`
`npm start`	Executar `scripts.start`
`npm run <name>`	Executar qualquer script personalizado
`pre<name>`	Executado automaticamente antes de `<name>`
`post<name>`	Executado automaticamente após `<name>`
`npm run`	Listar todos os scripts disponíveis

package.json

Inicializar e Campos

```
npm init                             # interactive setup
npm init -y                          # accept all defaults
npm pkg set name="my-app"            # set a field
npm pkg get version                  # read a field
```

Campos Principais

`name`	Nome do pacote (minúsculas, sem espaços)
`version`	Versão atual (semver: major.minor.patch)
`main`	Ponto de entrada para CommonJS (`require`)
`module`	Ponto de entrada para ES modules (bundlers)
`type`	`"module"` para ESM, `"commonjs"` para CJS (padrão)
`scripts`	Comandos nomeados (`build`, `test`, `start`, etc.)
`dependencies`	Dependências de produção
`devDependencies`	Dependências apenas de desenvolvimento
`engines`	Intervalos de versões requeridas de Node/npm

Versionamento

Comandos de Versão

```
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                    # set explicit version
npm version prerelease --preid=beta # 1.0.0-beta.0
```

Intervalos Semver

`^1.2.3`	Compatível: >=1.2.3 <2.0.0 (padrão)
`~1.2.3`	Nível de patch: >=1.2.3 <1.3.0
`1.2.3`	Somente versão exata
`>=1.0.0 <2.0.0`	Intervalo explícito
`*`	Qualquer versão
`1.x / 1.2.x`	Intervalos com curinga
`latest`	Tag da versão publicada mais recente

Publicação

Fluxo de Publicação

```
npm login                            # authenticate to registry
npm publish                          # publish public package
npm publish --access public          # scoped package as public
npm unpublish pkg@1.0.0              # remove specific version
npm deprecate pkg@"<2" "Use v2+"     # deprecate old versions
```

Referência de Publicação

`npm login`	Autenticar com o registro npm
`npm publish`	Publicar pacote no registro
`npm pack`	Criar tarball sem publicar
`npm unpublish`	Remover versão publicada (dentro de 72h)
`npm deprecate`	Marcar versões como obsoletas
`.npmignore`	Arquivos a excluir do pacote publicado
`files (package.json)`	Lista de arquivos a incluir no pacote

Workspaces

Comandos de Workspace

```
npm init -w packages/core            # create workspace
npm install -w packages/core lodash  # install in workspace
npm run build --workspaces           # run in all workspaces
npm run test -w packages/api         # run in specific workspace
npm ls --workspaces                  # list workspace deps
```

Configuração de Workspace

`workspaces (package.json)`	Array de globs de workspace: `["packages/*"]`
`-w / --workspace`	Apontar para um workspace específico
`--workspaces`	Executar comando em todos os workspaces
`--include-workspace-root`	Incluir pacote raiz nas operações de workspace
`npm install (root)`	Instala todas as dependências de workspace
`Hoisting`	Dependências compartilhadas movidas para node_modules raiz

npx

Executando com npx

```
npx create-react-app my-app          # run without installing
npx tsc --init                       # run local or remote bin
npx -p typescript tsc file.ts        # specify package explicitly
npx --yes create-next-app            # skip install prompt
npx node@18 -e "console.log('hi')" # run with specific Node
```

Opções do npx

`npx cmd`	Executar cmd de node_modules/.bin local ou remoto
`npx -p pkg cmd`	Instalar pkg e executar cmd
`npx --yes cmd`	Confirmar instalação automaticamente
`npx --no cmd`	Recusar instalação — falha se não estiver local
`npx -c 'cmd'`	Executar comando shell com PATH do npx
`npx node@ver`	Executar versão específica do Node.js

Configuração

Comandos de Configuração

```
npm config list                      # show current config
npm config set registry https://r.npmjs.com/
npm config set init-author-name "Name"
npm config get prefix                # global install path
npm config delete key                # remove a config value
```

Referência de Configuração

`.npmrc (project)`	Arquivo de configuração por projeto
`~/.npmrc`	Arquivo de configuração por usuário
`registry`	URL do registro de pacotes
`save-exact`	`true` para fixar versões exatas na instalação
`engine-strict`	`true` para aplicar campo `engines`
`fund`	`false` para suprimir mensagens de financiamento
`audit`	`false` para pular auditoria na instalação

Padrões Comuns

One-Liners

```
npm ls --depth=0 --json | jq '.dependencies | keys[]'
npm outdated --long                  # show type and homepage
npm cache clean --force              # clear npm cache
npm explain pkg                      # why is pkg installed?
npm exec -- envinfo --system         # system info for bug reports
```

Receitas

`Somente lock file`	`npm ci` — instalação limpa do package-lock.json
`Verificar licenças`	`npx license-checker --summary`
`Encontrar dependências não usadas`	`npx depcheck`
`Tamanho do bundle`	`npx bundlephobia-cli pkg` — verificar tamanho do pacote
`Atualizar tudo`	`npx npm-check-updates -u && npm install`
`Registro local`	`npx verdaccio` — executar registro privado