

Node.js

npm



Node.js

npm

Partie 1



NPM : GESTIONNAIRE DE PACKAGE

- > npm est le gestionnaire de package pour Node.js
 - ▶ Node Package Manager
 - ▶ Fonctionne en ligne de commande
 - ▶ Équivalent de Composer pour PHP
- > npmjs.com : répertoire des packages disponibles en Open Source
 - ▶ Possibilité de package privés en version payante
- > Permet d'initier un projet Node.js

INITIER UN PROJET NODE.JS

> En ligne de commande

- ▶ Créer un dossier **npm-demo**
- ▶ Se placer à l'intérieur du dossier
- ▶ Taper :

```
1 npm init
```

▶ Répondre aux questions (souvent en tapant Entrée)

- ☐ Nom de l'application
- ☐ Version
- ☐ Description
- ☐ Fichier principal
- ☐ Commande de test
- ☐ Dépôt GIT
- ☐ Mots-clés
- ☐ Auteur
- ☐ Type de licence

INITIER UN PROJET NODE.JS

- > Une fois toutes les questions répondues, npm présente le contenu du fichier `package.json` qu'il est sur le point de créer. Valider la création.

Exécution de npm init :

```
1  package name: (npm-demo)
2  version: (1.0.0)
3  description:
4  entry point: (index.js)
5  test command:
6  git repository:
7  keywords:
8  author:
9  license: (ISC)
10 About to write to /Volumes/Data/3iL/www/b3_nodejs/npm-demo/package.json:
11
12 {
13   "name": "npm-demo",
14   "version": "1.0.0",
15   "description": "",
16   "main": "index.js",
17   "scripts": {
18     "test": "echo \"Error: no test specified\" && exit 1"
19   },
20   "author": "",
21   "license": "ISC"
22 }
23
24
25 Is this OK? (yes)
```

PACKAGE.JSON

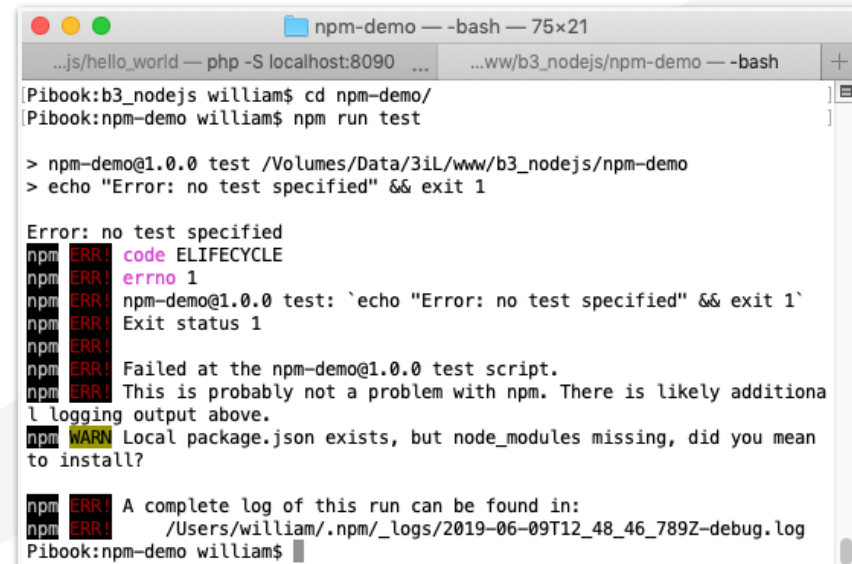
- > Contenu du fichier `package.json`

package.json :

```
1  {
2    "name": "npm-demo",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "test": "echo \"Error: no test specified\" && exit 1"
8    },
9    "author": "",
10   "license": "ISC"
11 }
```

Commandes exécutables via `npm run`

- > Zone scripts permet de définir des commandes à exécuter
- > Taper `npm run test`



```
npm-demo — -bash — 75x21
...js/hello_world — php -S localhost:8090 ...
...ww/b3_nodejs/npm-demo — -bash
[Pibook:b3_nodejs william$ cd npm-demo/
[Pibook:npm-demo william$ npm run test

> npm-demo@1.0.0 test /Volumes/Data/3iL/www/b3_nodejs/npm-demo
> echo "Error: no test specified" && exit 1

Error: no test specified
npm ERR! code ELIFECYCLE
npm ERR! errno 1
npm ERR! npm-demo@1.0.0 test: `echo "Error: no test specified" && exit 1`
npm ERR! Exit status 1
npm ERR!
npm ERR! Failed at the npm-demo@1.0.0 test script.
npm ERR! This is probably not a problem with npm. There is likely additional
logging output above.
npm WARN Local package.json exists, but node_modules missing, did you mean
to install?

npm ERR! A complete log of this run can be found in:
npm ERR! /Users/william/.npm/_logs/2019-06-09T12_48_46_789Z-debug.log
Pibook:npm-demo william$
```

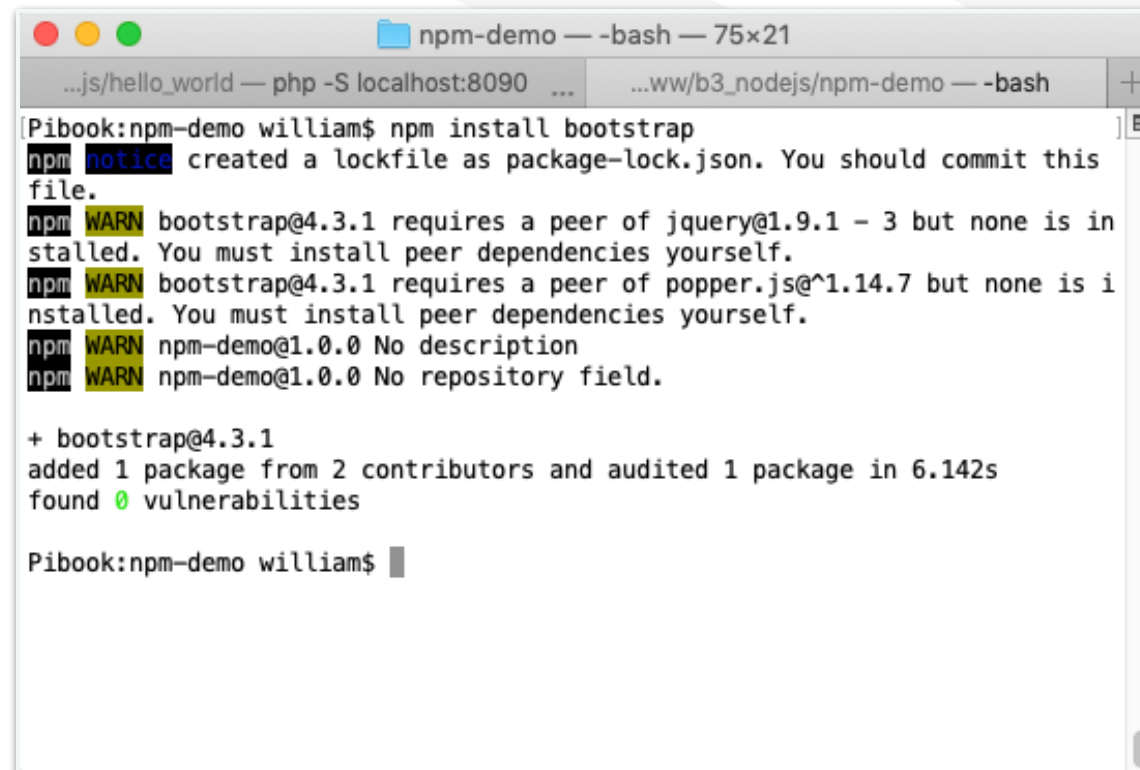
INSTALLER UN PACKAGE

- > Pour installer un package, il suffit de faire :

`npm install monPackage`

- > Exemple installer Bootstrap :

`npm install bootstrap`



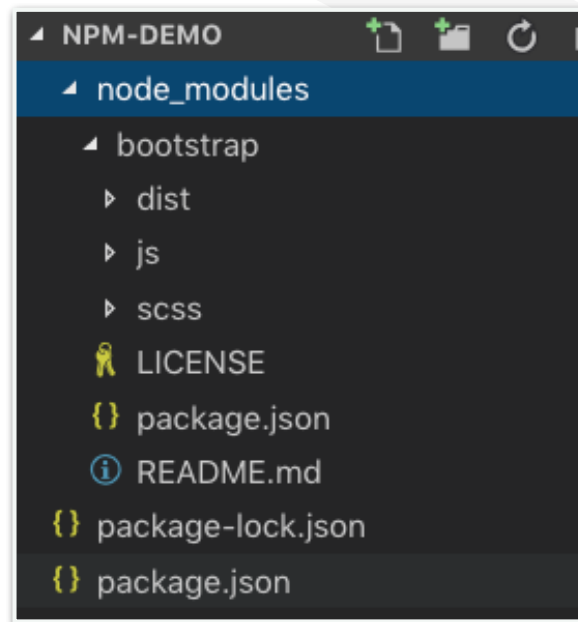
```
npm-demo — -bash — 75x21
...js/hello_world — php -S localhost:8090 ...
...ww/b3_nodejs/npm-demo — -bash
Pibook:npm-demo william$ npm install bootstrap
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN bootstrap@4.3.1 requires a peer of jquery@1.9.1 - 3 but none is installed. You must install peer dependencies yourself.
npm WARN bootstrap@4.3.1 requires a peer of popper.js@^1.14.7 but none is installed. You must install peer dependencies yourself.
npm WARN npm-demo@1.0.0 No description
npm WARN npm-demo@1.0.0 No repository field.

+ bootstrap@4.3.1
added 1 package from 2 contributors and audited 1 package in 6.142s
found 0 vulnerabilities

Pibook:npm-demo william$
```

INSTALLER UN PACKAGE

- > Cela a eu pour effet :
 - ▶ La modification du fichier **package.json** pour intégrer la dépendance
 - ▶ La création du fichier **package-lock.json** pour garder trace de la version exacte installée.
 - ▶ La création du dossier **node_modules** pour stocker de façon organisée les dépendances du projet.



INSTALLER UN PACKAGE

- > Contenu du fichier `package.json` :

`package.json` :

```
1  {
2    "name": "npm-demo",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "test": "echo \"Error: no test specified\" && exit 1"
8    },
9    "author": "",
10   "license": "ISC",
11   "dependencies": {
12     "bootstrap": "^4.3.1"
13   }
14 }
```

liste des dépendances du projet

- > Existence d'une option `--save` que l'on peut encore trouver dans d'ancienne documentation. Avant npm 5.0 :
 - ▶ `--save` : enregistre la dépendance dans `package.json` (inutile depuis)
 - ▶ Sans : installe la dépendance sans la faire figurer dans `package.json`

Node.js

npm

Partie 2



LES NUMÉROS DE VERSION

- > Codage sur trois nombres des numéros de versions

4.3.1

Numéro majeur
de version

Numéro mineur
de version

Numéro de patch

- > Numéro majeur : "grand saut de version"
- > Numéro mineur : ajout de fonctionnalité sans remettre en cause le package
- > Numéro de patch : correction d'erreur

LES NUMÉROS DE VERSION

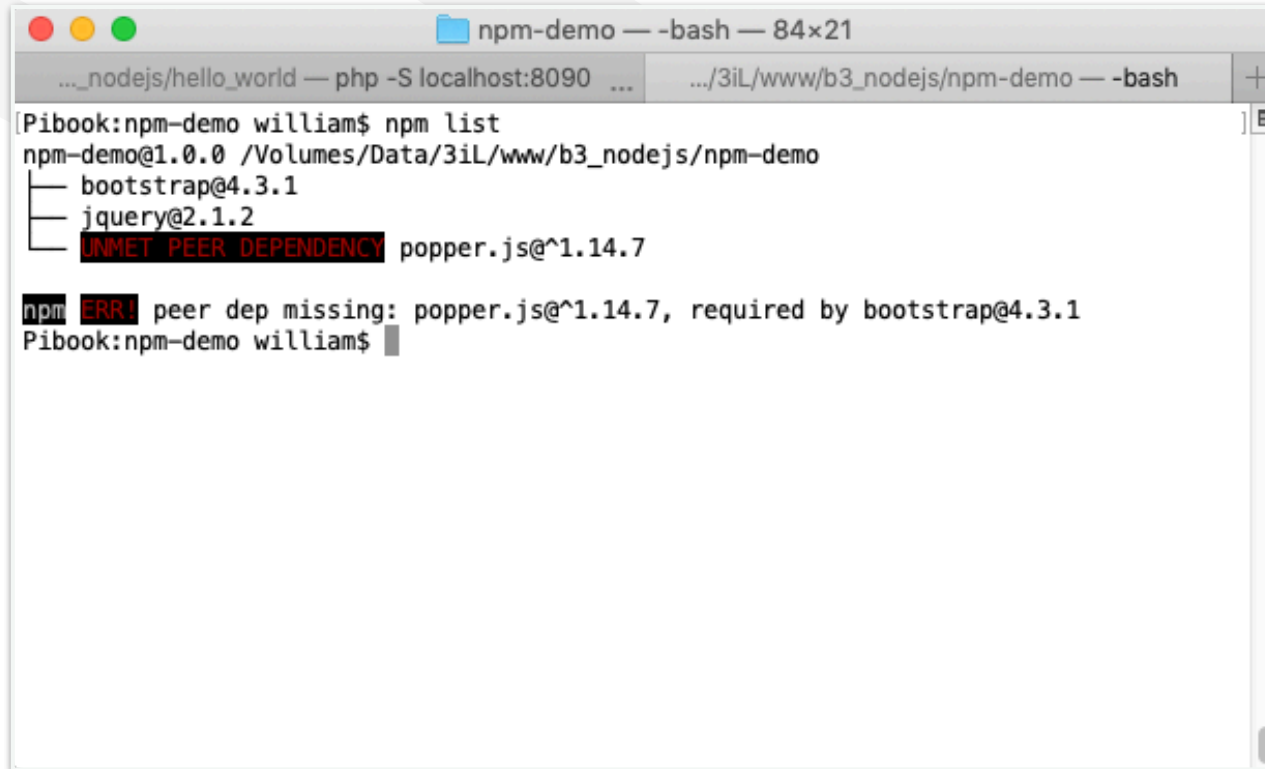
- > ^ et ~ sont les deux principaux symboles rencontrés dans les numéros de versions avec npm.
- > Exemple avec Bootstrap (4.3.1)
 - ▶ ^4.3.1 : installera n'importe quelle version 4.x.x, ne prendra pas en compte la version 5.0.0
 - ▶ ~4.3.1 : installera n'importe quelle version 4.3.x, ne prendra pas en compte la 4.4.0
- > Sans aucun symbole : 4.3.1 npm n'installera que cette version précisément.

LES NUMÉROS DE VERSION

- > Utiliser @ pour installer une version précise d'une dépendance :
`npm install jquery@2.1.2`
- > Attention :
 - ▶ npm installera bien la version 2.1.2
 - ▶ npm inscrira ^2.1.2 dans le fichier `package.json`
- > Pour installer la dernière des versions 2 de jquery :
`npm install jquery@2`
- > Pour installer la dernière des versions 2.1 de jquery :
`npm install jquery@2.1`
- > Note : actuellement jquery est en version 3.4.1

LISTER LES DÉPENDANCE

- > Pour lister les dépendances : `npm list`



```
npm-demo — -bash — 84x21
..._nodejs/hello_world — php -S localhost:8090 ...
.../3iL/www/b3_nodejs/npm-demo — -bash
Pibook:npm-demo william$ npm list
npm-demo@1.0.0 /Volumes/Data/3iL/www/b3_nodejs/npm-demo
├── bootstrap@4.3.1
├── jquery@2.1.2
└── UNMET PEER DEPENDENCY popper.js@^1.14.7

npm ERR! peer dep missing: popper.js@^1.14.7, required by bootstrap@4.3.1
Pibook:npm-demo william$
```

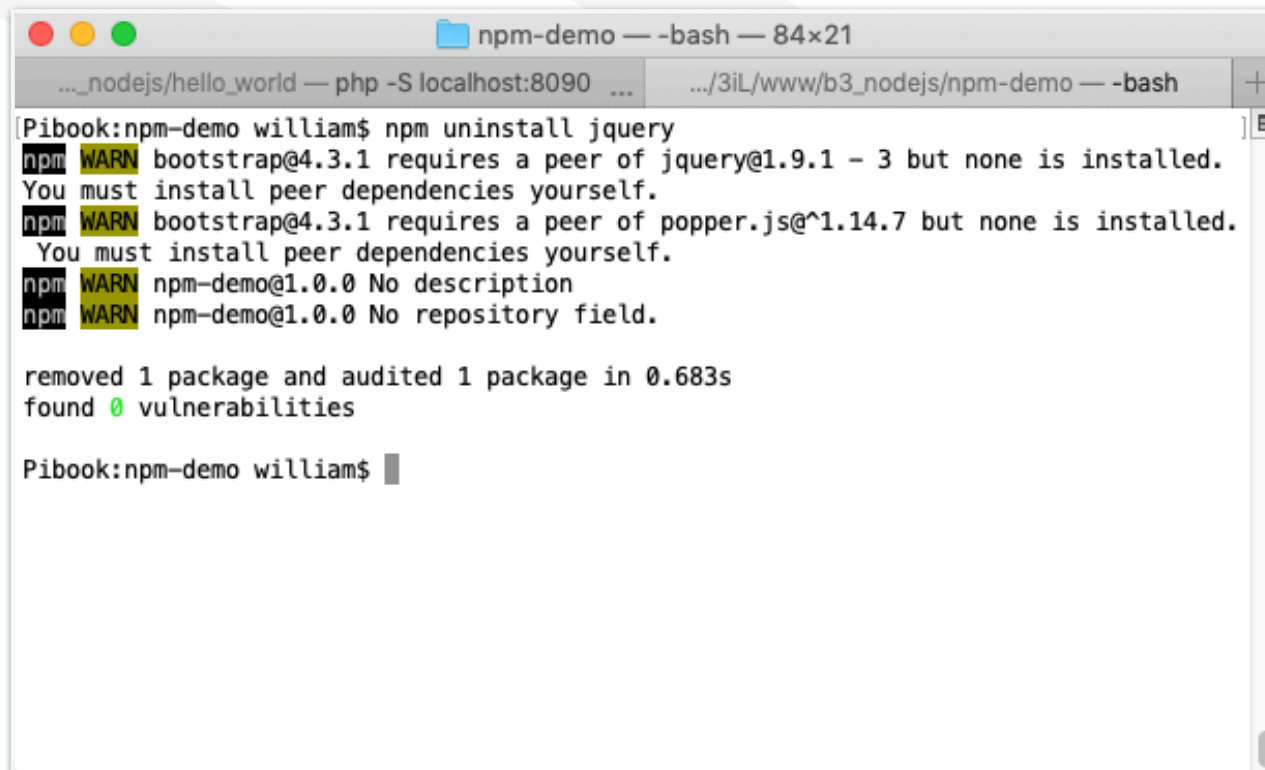
- > Note : ici `popper.js` est déclaré manquant pour Bootstrap (non obligatoire)

RÉINSTALLER TOUTES LES DÉPENDANCES

- > `npm install` permet de réinstaller toutes les dépendances
 - ▶ Supprimer le dossier `node_modules`
 - ▶ Taper `npm install` et toutes les dépendances seront réinstallées
- > Fonctionnement avec git
 - ▶ Dépendances inutiles à versionner dans un projet
 - ▶ Mettre le dossier `node_modules` dans `.gitignore`
 - ▶ À la récupération d'un projet un simple `npm install` permet de retrouver le projet complet
 - ▶ Évidemment : le fichier `package.json` est ultra important.

SUPPRIMER UNE DÉPENDANCE

- > Pour supprimer une dépendance il suffit d'exécuter `npm uninstall` :
`npm uninstall jquery`



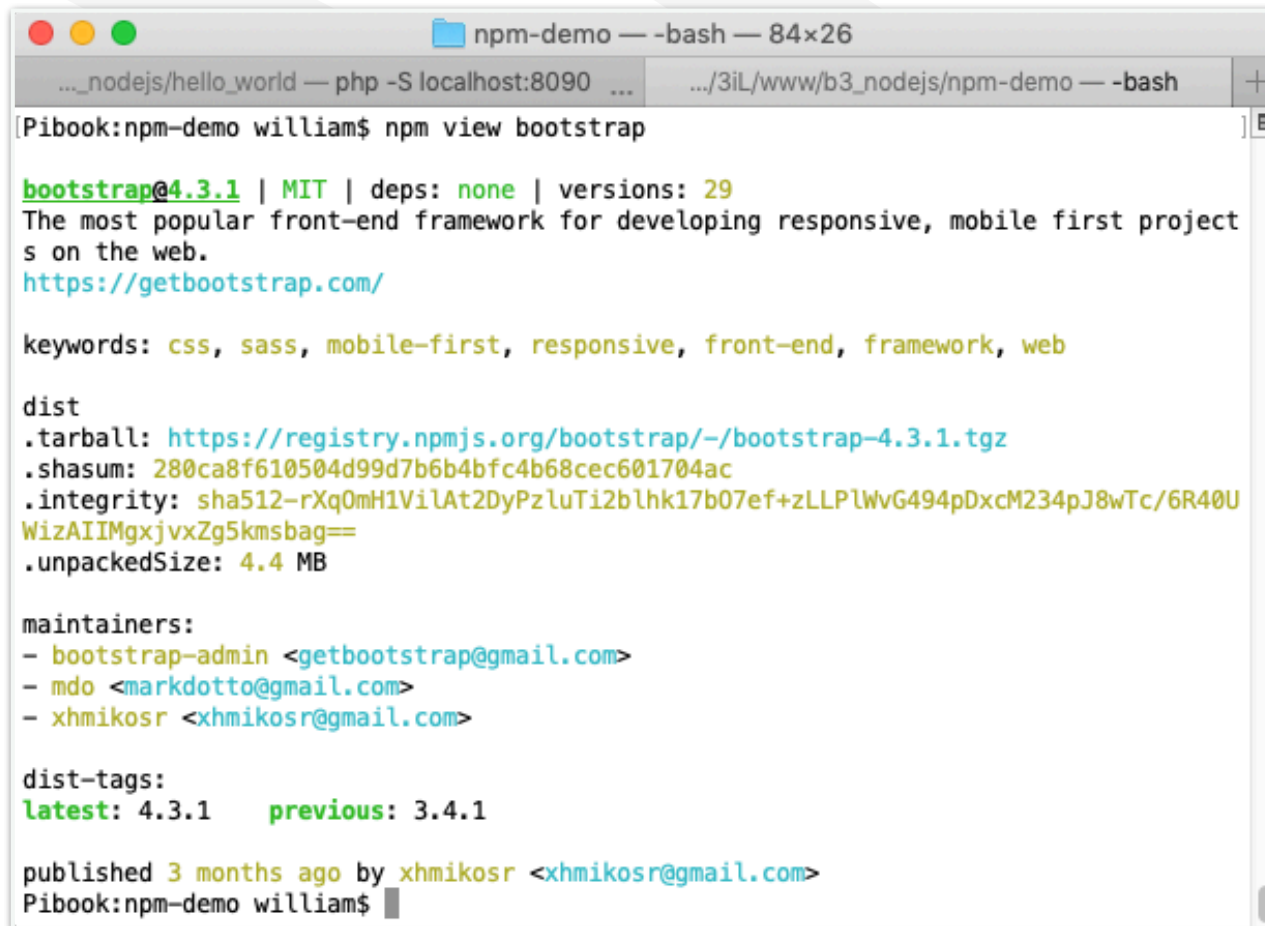
```
npm-demo — -bash — 84x21
..._nodejs/hello_world — php -S localhost:8090 ...  .../3iL/www/b3_nodejs/npm-demo — -bash
Pibook:npm-demo william$ npm uninstall jquery
npm WARN bootstrap@4.3.1 requires a peer of jquery@1.9.1 - 3 but none is installed.
You must install peer dependencies yourself.
npm WARN bootstrap@4.3.1 requires a peer of popper.js@^1.14.7 but none is installed.
You must install peer dependencies yourself.
npm WARN npm-demo@1.0.0 No description
npm WARN npm-demo@1.0.0 No repository field.

removed 1 package and audited 1 package in 0.683s
found 0 vulnerabilities

Pibook:npm-demo william$
```


INFORMATIONS SUR UN PACKAGE

- > Pour avoir des informations sur un package `npm view` :
`npm view bootstrap`

A terminal window titled 'npm-demo — -bash — 84x26' showing the output of the command 'npm view bootstrap'. The output displays metadata for the bootstrap package, including its version (4.3.1), description, keywords, distribution information, maintainers, and tags.

```
Pibook:npm-demo william$ npm view bootstrap

bootstrap@4.3.1 | MIT | deps: none | versions: 29
The most popular front-end framework for developing responsive, mobile first project
s on the web.
https://getbootstrap.com/

keywords: css, sass, mobile-first, responsive, front-end, framework, web

dist
.tarball: https://registry.npmjs.org/bootstrap/-/bootstrap-4.3.1.tgz
.shasum: 280ca8f610504d99d7b6b4bfc4b68cec601704ac
.integrity: sha512-rXq0mH1Vi1At2DyPzluTi2blhk17b07ef+zLLPlWvG494pDxcM234pJ8wTc/6R40U
WizAIIMgxjvxZg5kmsbag==
.unpackedSize: 4.4 MB

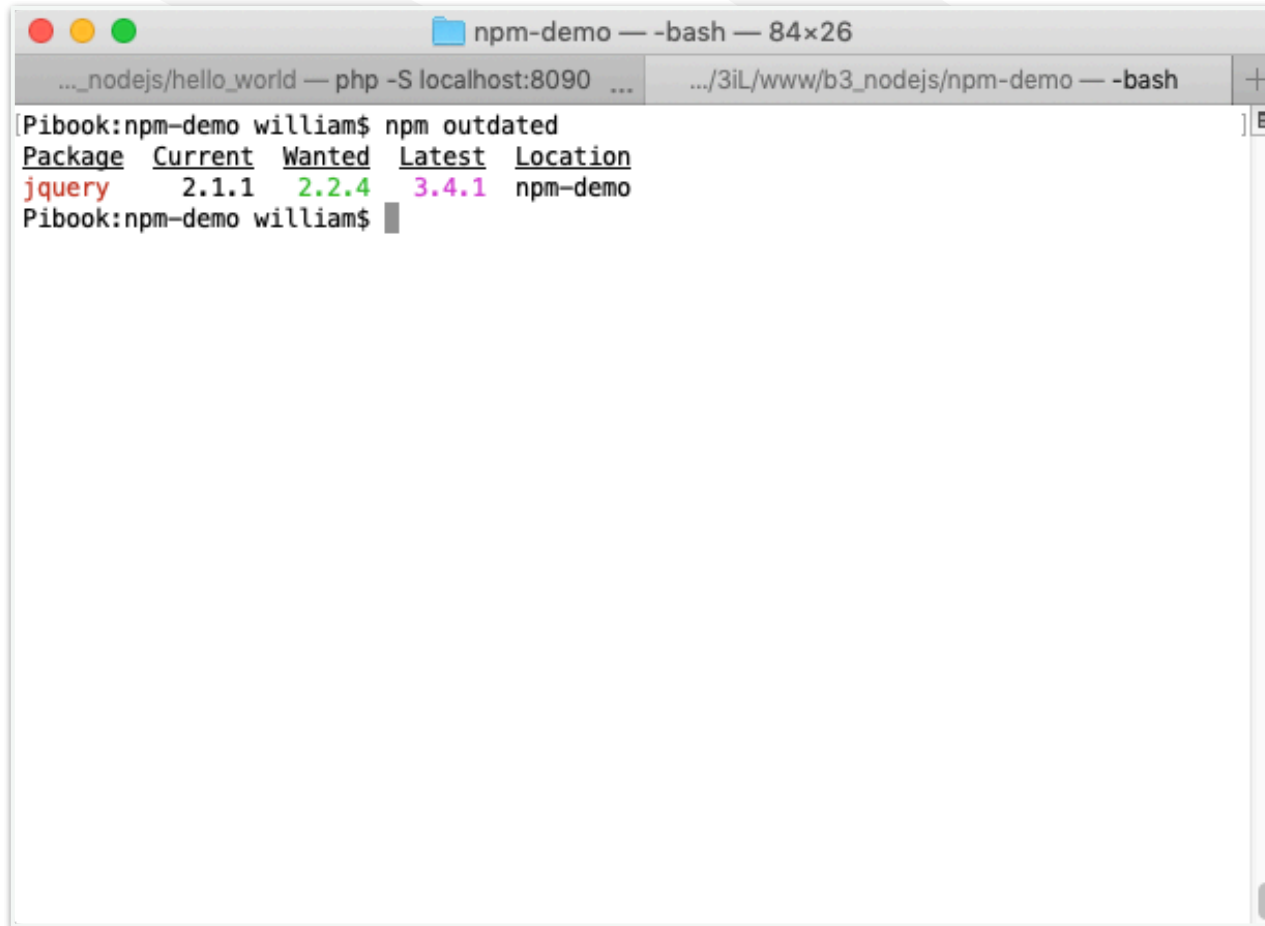
maintainers:
- bootstrap-admin <getbootstrap@gmail.com>
- mdo <markdotto@gmail.com>
- xhmikosr <xhmikosr@gmail.com>

dist-tags:
latest: 4.3.1   previous: 3.4.1

published 3 months ago by xhmikosr <xhmikosr@gmail.com>
Pibook:npm-demo william$
```

METTRE À JOUR LES DÉPENDANCES

- > Installer la version 2.1.1 de jQuery
- > Pour lister les dépendances qui peuvent être mises à jour :
`npm outdated`

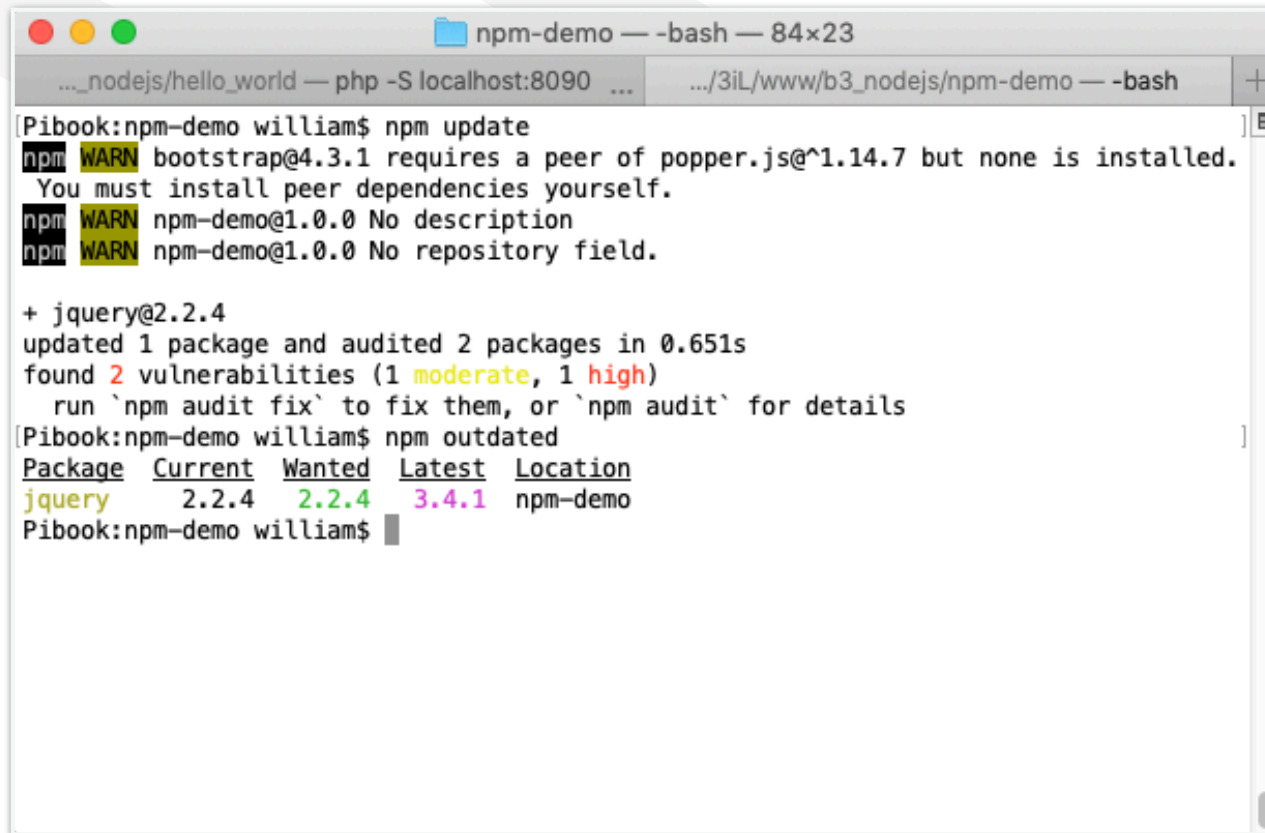


```
npm-demo — -bash — 84x26
..._nodejs/hello_world — php -S localhost:8090 ...
.../3iL/www/b3_nodejs/npm-demo — -bash
Pibook:npm-demo william$ npm outdated
Package  Current  Wanted  Latest  Location
jquery   2.1.1    2.2.4   3.4.1   npm-demo
Pibook:npm-demo william$
```

The image shows a terminal window titled 'npm-demo — -bash — 84x26'. The prompt is 'Pibook:npm-demo william\$'. The command 'npm outdated' has been executed, resulting in a table of outdated packages. The table has five columns: 'Package', 'Current', 'Wanted', 'Latest', and 'Location'. One package, 'jquery', is listed with a current version of 2.1.1, a wanted version of 2.2.4, and a latest version of 3.4.1, located at 'npm-demo'.

METTRE À JOUR LES DÉPENDANCES

> Taper `npm update` :



```
npm-demo — -bash — 84x23
..._nodejs/hello_world — php -S localhost:8090 ...  .../3iL/www/b3_nodejs/npm-demo — -bash
Pibook:npm-demo william$ npm update
npm WARN bootstrap@4.3.1 requires a peer of popper.js@^1.14.7 but none is installed.
  You must install peer dependencies yourself.
npm WARN npm-demo@1.0.0 No description
npm WARN npm-demo@1.0.0 No repository field.

+ jquery@2.2.4
updated 1 package and audited 2 packages in 0.651s
found 2 vulnerabilities (1 moderate, 1 high)
  run `npm audit fix` to fix them, or `npm audit` for details
Pibook:npm-demo william$ npm outdated
Package Current Wanted Latest Location
jquery  2.2.4  2.2.4  3.4.1  npm-demo
Pibook:npm-demo william$
```

DÉPENDANCES DE DÉVELOPPEMENT

- > Certaines dépendances peuvent ne concerner que la phase de développement :
 - ▶ Aucun intérêt à être présentes en production
 - ▶ Principe : tout ce qui n'est pas utile peut être dangereux
 - ▶ Exemple : tout ce qui est relatif aux tests ou à faciliter la vie lors du développement
- > Exemple : **nodemon**
 - ▶ Package servant à relancer automatiquement l'application en cas de modification de ses sources
 - ▶ N'a pas de raison d'être pour la version en production de l'application
- > npm permet de séparer ce type de dépendances en ajoutant **--save-dev** lors de leur installation

DÉPENDANCES DE DÉVELOPPEMENT

- > Taper : `npm install nodemon --save-dev`
- > Observer le contenu du fichier `package.json`

package.json :

```
1  {
2    "name": "npm-demo",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "test": "echo \"Error: no test specified\" && exit 1"
8    },
9    "author": "",
10   "license": "ISC",
11   "dependencies": {
12     "bootstrap": "^4.3.1",
13     "jquery": "^2.2.4"
14   },
15   "devDependencies": {
16     "nodemon": "^1.19.1"
17   }
18 }
19
```

liste des dépendances du développement du projet

DÉPENDANCES DE DÉVELOPPEMENT

> Au passage, faire un `npm list` pour se rendre compte que la liste s'est étoffée :



```
npm-demo — -bash — 84x38
..._nodejs/hello_world — php -S localhost:8090 ...
.../3iL/www/b3_nodejs/npm-demo — -bash
[Pibook:npm-demo william$ npm list
npm-demo@1.0.0 /Volumes/Data/3iL/www/b3_nodejs/npm-demo
├── bootstrap@4.3.1
├── jquery@2.2.4
├── nodemon@1.19.1
├── chokidar@2.1.6
├── anymatch@2.0.0
├── micromatch@3.1.10
├── arr-diff@4.0.0
├── array-unique@0.3.2 deduped
├── braces@2.3.2 deduped
├── define-property@2.0.2
├── is-descriptor@1.0.2
├──   is-accessor-descriptor@1.0.0
├──   kind-of@6.0.2 deduped
├──   is-data-descriptor@1.0.0
├──   kind-of@6.0.2 deduped
├──   kind-of@6.0.2 deduped
├── isobject@3.0.1 deduped
├── extend-shallow@3.0.2
├── assign-symbols@1.0.0
├── is-extendable@1.0.1
├──   is-plain-object@2.0.4
├──   isobject@3.0.1 deduped
├── extglob@2.0.4
├──   array-unique@0.3.2 deduped
├──   define-property@1.0.0
├──   is-descriptor@1.0.2
├──   is-accessor-descriptor@1.0.0
├──   kind-of@6.0.2 deduped
├──   is-data-descriptor@1.0.0
├──   kind-of@6.0.2 deduped
├──   kind-of@6.0.2 deduped
├── expand-brackets@2.1.4
├── debug@2.6.9
├── ms@2.0.0 deduped
├── define-property@0.2.5
├── is-descriptor@0.1.6 deduped
```

Node.js

npm

Partie 3



UTILISER NODEMON

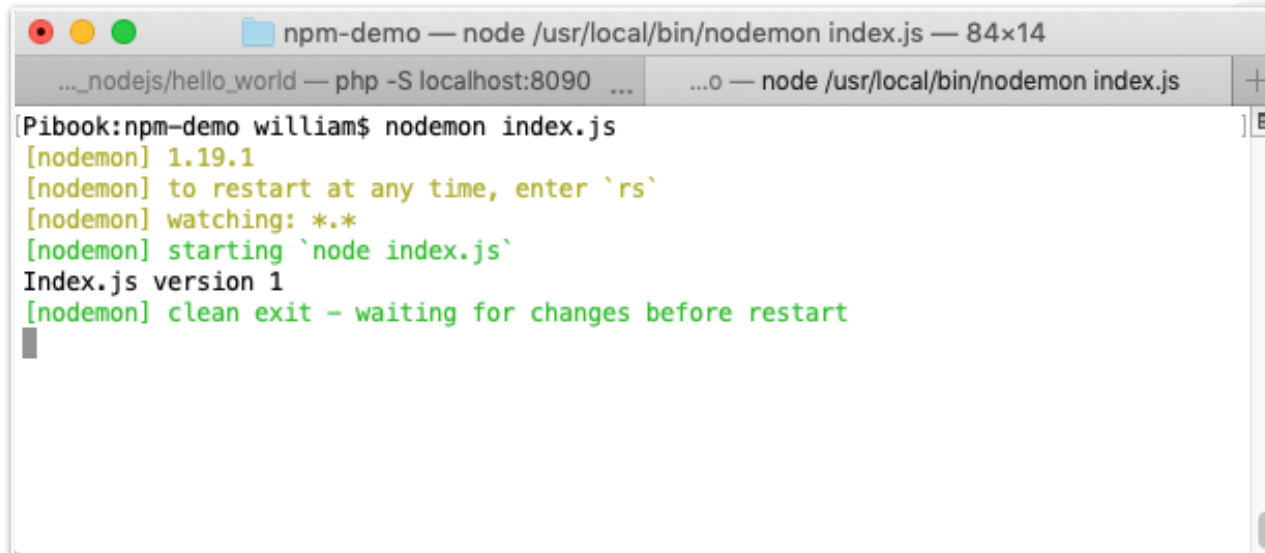
- > Pour utiliser nodemon, nous allons :
 - ▶ Créer un fichier **index.js** simple (affichage)
 - ▶ Démarrer le projet avec nodemon
 - ▶ Modifier le fichier index.js

- > Créer le fichier **index.js** comme suit :

package.json :

```
1 console.log("Index.js version 1");
```

- > Taper **nodemon index.js**

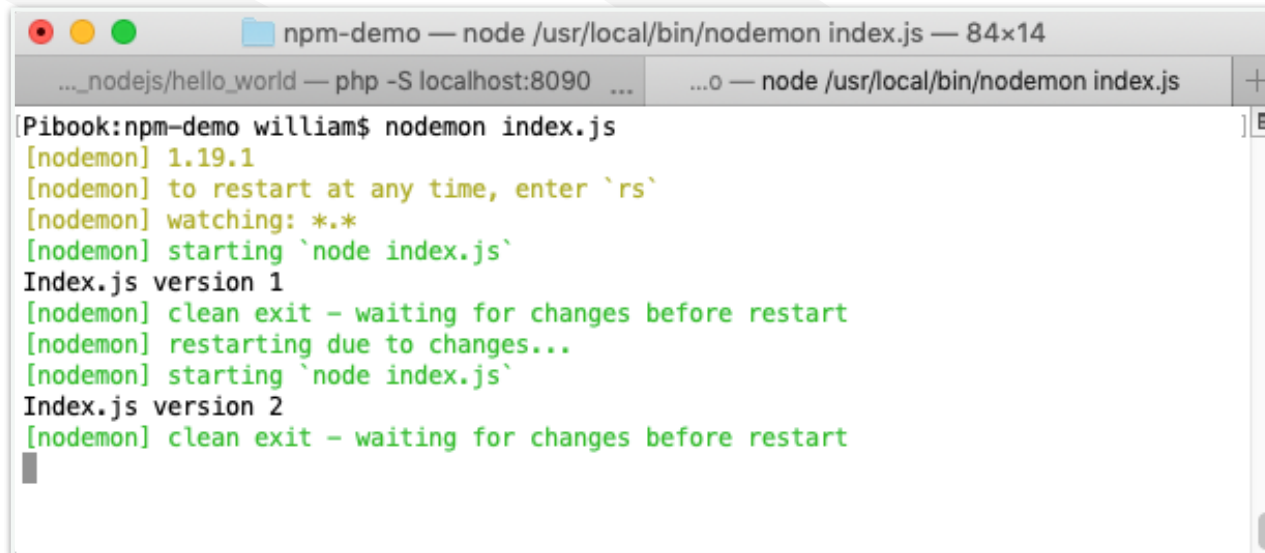


A terminal window titled "npm-demo — node /usr/local/bin/nodemon index.js — 84x14". The terminal shows the command "Pibook:npm-demo william\$ nodemon index.js" and the following output: "[nodemon] 1.19.1", "[nodemon] to restart at any time, enter `rs`", "[nodemon] watching: *.*", "[nodemon] starting `node index.js`", "Index.js version 1", and "[nodemon] clean exit - waiting for changes before restart".

```
Pibook:npm-demo william$ nodemon index.js
[nodemon] 1.19.1
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node index.js`
Index.js version 1
[nodemon] clean exit - waiting for changes before restart
```


UTILISER NODEMON

- > Modifier le fichier `index.js` pour passer "à la version 2".
- > Automatiquement `index.js` sera relancé dans le terminal.



```
npm-demo — node /usr/local/bin/nodemon index.js — 84x14
..._nodejs/hello_world — php -S localhost:8090 ...  ...o — node /usr/local/bin/nodemon index.js
Pibook:npm-demo william$ nodemon index.js
[nodemon] 1.19.1
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node index.js`
Index.js version 1
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
Index.js version 2
[nodemon] clean exit - waiting for changes before restart
```

- > Nodemon scan en permanence les modifications dans le dossier.
- > Même la modification d'un fichier qui n'intervient pas directement provoquera la relance de `index.js`.
- > C'est particulièrement pratique pour relancer un programme serveur automatiquement.

UTILISER LA RUBRIQUE SCRIPTS

- > La rubrique `scripts` de `package.json` peut être enrichie de nos propres scripts.

`package.json` :

```
1  {
2    "name": "npm-demo",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "test": "echo \"Error: no test specified\" && exit 1",
8      "start": "nodemon index.js"
9    },
10   "author": "",
11   "license": "ISC",
12   "dependencies": {
13     "bootstrap": "^4.3.1",
14     "jquery": "^2.2.4"
15   },
16   "devDependencies": {
17     "nodemon": "^1.19.1"
18   }
19 }
```

- > Il suffit maintenant de faire `npm run start` au lieu de `nodemon index.js`
- > Comme `start` est souvent utilisée, on peut même taper `npm start` (cas particulier)

INSTALLATION GLOBALE

- > Par défaut, npm réalise l'installation dans le dossier local.
- > Possibilité de faire une installation "globale" (centralisée)
 - ▶ Utile surtout pour des outils comme Nodemon, LESS ou SASS
 - ▶ Permet d'enrichir le terminal de nouvelles commandes disponibles de partout
 - ▶ N'a pas forcément de sens pour tous les packages
- > Pour faire une installation globale de LESS
`npm install -g less`
- > Ne modifie pas le fichier `package.json`
- > Attention : il faut avoir les droits pour que cela fonctionne.
- > La commande `less` doit être disponible dans le terminal.
`less --v`