

Introducció a npm



Continguts

1. Què és npm?	3
2. Creació d'un paquet amb node	3
3. Execució del paquet	3
4. Instal·lació de paquets	4
5. Altres ordres d'utilitat	6

1. Què és npm?

Npm (Node Package Manager) és el gestor de paquets de *node*, o mòduls de javascript, tant del costat del client (FrontEnd) com del servidor (BackEnd). Salvant les distàncies, *npm* vindria a ser per a Javascript com *Maven* o *Gradle* per a Java.

Per tal de veure la versió que tenim instal·lada de *npm*, podem consultar-ho amb:

```
1 $ npm -v
2 6.13.3
```

2. Creació d'un paquet amb node

Podem entendre un paquet nodejs com un programa descrit per un fitxer **package.json**, que vindria a ser com el fitxer **build.gradle** dels paquets de Gradle.

Per tal de crear un paquet de zero, fem ús de l'ordre **npm init**, que llançarà un xicotet assistent per tal de crear aquest fitxer **package.json**.

Aquest assistent ens demanarà informació com el nom del paquet (per defecte el de la carpeta on estem), la versió d'aquest, el fitxer d'entrada (entry point), l'autor o la llicència entre d'altres. Aquest assistent només fa que crear-nos el fitxer **package.json**, amb la informació que li hem proporcionat.

Un fitxer **package.json** acabat de crear amb l'assistent tindrà una estructura semblant a la següent:

```
1 {
2   "name": "provajs",
3   "version": "1.0.0",
4   "description": "Paquet de prova per a nodejs",
5   "main": "index.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "author": "",
10  "license": "ISC"
11 }
```

Com a dades a destacar, tenim el nom i la versió del paquet, la descripció i el fitxer principal de l'aplicació.

3. Execució del paquet

Si creem un poc de contingut, com per exemple un *Hola mon* al fitxer **index.js**:

```
1 console.log("Hola món");
```

Per executar-lo, com ja hem vist als documents d'introducció a node, farem:

```
1 nodejs index.js
```

Amb *npm*, tot i que hem indicat al camp *main* quin és el fitxer original, no podem executar-lo directament. Per fer ús de *npm* per llançar el nostre paquet, caldrà afegir un xicotet script d'execució. Per a això, modifiquem l'apartat d'*scripts* de la següent forma:

```
1 "scripts": {  
2   "test": "echo \"Error: no test specified\" && exit 1",  
3   "start": "node index.js"  
4 },
```

Com veiem, hem afegit un script, anomenat *start* que el que farà serà llançar per nosaltres el *node index.js*. Amb açò, podrem llançar el nostre paquet amb *npm* de la següent forma:

```
1 npm start
```

En aquest cas, podriem utilitzar de forma indiferent *npm start* o bé *node index.js*. En altres exemples que vorem més endavant, per exemple, quan utilitzem *electron*, ens resultarà més còmode llançar l'execució a través de *npm*.

4. Instal·lació de paquets

Podem buscar paquets al lloc web <https://www.npmjs.com/>, que ve a ser com el repositori de Maven per a Java. A més, des de la línia d'ordres també podem utilitzar l'ordre *npm search* per buscar paquets pel seu nom.

Els paquets poden instal·lar-se de diferents formes:

- **De forma global** al sistema operatiu; pensat per a paquets que anem a utilitzar com a aplicacions i ens van a ser d'utilitat per als nostres projectes: compiladors CSS, generadors de projectes, etc. Aquesta instal·lació s'ha de fer com a *sudo*, de la següent forma:

```
1 $ sudo npm install nom_del_paquet -g
```

Per exemple, anem a utilitzar *npm* per instal·lar/actualitzar el propi *npm*:

```
1 $ sudo npm install -g npm
```

Si ara tornem a comprovar la versió (caldrà fer-ho amb una altra terminal):

```
1 $ npm -v
2 6.14.9
```

- Instal·lació de paquets a **nivell de projecte**, és a dir, com a dependència d'aquest. A més, dins el projecte, es pot especificar de dues formes:
 - Com a **dependència de desenvolupament i producció**:

```
1 $ npm install nom_del_modul --save
```

- Com a **dependència de desenvolupament**, és a dir, ens servirà per construir el projecte, però no es necessita quan aquest s'està executant:

```
1 $ npm install nom_del_modul --save-dev
```

L'ordre `npm install` també pot utilitzar-se sense cap altre modificador. En eixe cas, per defecte interpretarà que es tracta d'una dependència de l'aplicació, afegint-la en la secció corresponent (com si ferem un `--save`)

Anem a veure un exemple. Instal·lem, per posar un exemple el paquet `xml`, per tractar fitxers d'aquest tipus. Per a això faríem, dins la carpeta on tenim el fitxer `package.json`:

```
1 $ npm install --save xml
```

Després d'açò, notarem dos canvis:

- En primer lloc, tindrem una carpeta `node_modules` al directori, on tindrem tots els paquets de què depenem. Ací haurà descarregat el paquet `xml` i tots els paquets que aquest mateix necessita.
- En segon lloc, i al haver utilitzat `--save` (fixeu-se que pot anar davant del nom del paquet, no necessàriament darrere), `npm` ens haurà modificat el fitxer `package.json` afegint el següent:

```
1 "dependencies": {
2   "xml": "^1.0.1"
3 }
```

Que com veiem, conté una entrada de dependències, amb el nom del paquet i la versió mínima que requereix.

Gràcies a aquest fitxer, podem tornar a generar la carpeta `node_modules` i descarregar les dependències necessàries. A l'igual que féiem amb les llibreries de Gradle, quan distribuïm la nostra aplicació, aquesta carpeta no s'ha d'incloure (si està en un repositori `git`, caldria posar-la al `.gitignore`), de manera que es pot reconstruir amb `npm install`. Aquesta ordre, revisarà el fitxer `package.json`, i instal·larà les dependències necessàries.

5. Altres ordres d'utilitat

Bàsicament, les ordres que hem vist més amunt seran les que més utilitzarem: `npm init` per generar projectes, i `npm install` per afegir dependències. De tota manera, `npm` suporta algunes ordres més. Podem consultar aquestes ordres amb:

```
1 $ npm --help
2 Usage: npm <command>
3
4 where <command> is one of:
5   access, adduser, audit, bin, bugs, c, cache, ci, cit,
6   clean-install, clean-install-test, completion, config,
7   create, ddp, dedupe, deprecate, dist-tag, docs, doctor,
8   edit, explore, fund, get, help, help-search, hook, i, init,
9   install, install-ci-test, install-test, it, link, list, ln,
10  login, logout, ls, org, outdated, owner, pack, ping, prefix,
11  profile, prune, publish, rb, rebuild, repo, restart, root,
12  run, run-script, s, se, search, set, shrinkwrap, star,
13  stars, start, stop, t, team, test, token, tst, un,
14  uninstall, unpublish, unstar, up, update, v, version, view,
15  whoami
16
17 npm <command> -h  quick help on <command>
18 npm -l            display full usage info
19 npm help <term>   search for help on <term>
20 npm help npm      involved overview
21
22 Specify configs in the ini-formatted file:
23   /home/joamuran/.npmrc
24 or on the command line via: npm <command> --key value
25 Config info can be viewed via: npm help config
```

El que hem vist fins ara és una introducció molt breu a *npm*, però ens serà suficient com a punt de partida per al curs. A Internet podeu trobar moltíssima informació sobre aquest mòdul. Si voleu ampliar un poc el què hem vist en aquest document, podeu començar amb l'article <https://www.mmfilesi.com/blog/introduccion-a-npm/>.