Introducció a npm

IES Jaume II El Just
Tavernes de la Valldigna

Aplicacions multiplataforma amb tecnologies web

Curs 20-21

Continguts

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 Javscript com *Maven* o *Gradle* per a Java.

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

```
$ npm -v
3.5.2
```

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 paques 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 estrutcura semblant a le següent:

```
{
  "name": "provajs",
  "version": "1.0.0",
  "description": "Paquet de prova per a nodejs",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}
```

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:

```
console.log("Hola món");
```

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

```
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:

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

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:

```
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 llençar 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, generados de projectes, etc. Aquesta instal·lació s'ha de fer com a sudo, de la següent forma:

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

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

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

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

```
$ npm -v
6.13.3
```

- 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**:

```
$ npm instal 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:

```
$ npm instal nom_del_modul --save-dev
```

L'ordre `npm install` també pot utilitzar-se sense cap altre modificador. En eixe cas, descarrega i instal·la el paquet, però no modifica el fitxer **`package`**`.json` indicant que aquest és una dependència.

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

```
$ 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. Aci 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:

```
 "dependencies": {
    "xml": "^1.0.1"
  }
```

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:

```
$ npm --help
Usage: npm <command>

where <command> is one of:
    access, adduser, audit, bin, bugs, c, cache, ci, cit,
    clean-install, clean-install-test, completion, config,
    create, ddp, dedupe, deprecate, dist-tag, docs, doctor,
    edit, explore, fund, get, help, help-search, hook, i, init,
    install, install-ci-test, install-test, it, link, list, ln,
    login, logout, ls, org, outdated, owner, pack, ping, prefix,
    profile, prune, publish, rb, rebuild, repo, restart, root,
    run, run-script, s, se, search, set, shrinkwrap, star,
    stars, start, stop, t, team, test, token, tst, un,
    uninstall, unpublish, unstar, up, update, v, version, view,
    whoami

npm <command> -h  quick help on <command>
npm -l            display full usage info
npm help <term>   search for help on <term>
npm help npm      involved overview

Specify configs in the ini-formatted file:
    /home/joamuran/.npmrc
or on the command line via: npm <command> --key value
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/.