Node.js - NPM

Node Package Manager (NPM) provides two main functionalities −

* Online repositories for node.js packages/modules
* Command line utility to install Node.js packages, do version management and dependency management of Node.js packages.

NPM comes bundled with Node.js installables after v0.6.3 version.

$ npm --version

2.7.1

If you are running an old version of NPM then update it to the latest version. Just use the following command from root −

$ sudo npm install npm -g

/usr/bin/npm -> /usr/lib/node\_modules/npm/bin/npm-cli.js

npm@2.7.1 /usr/lib/node\_modules/npm

## **Installing Modules using NPM**

There is a simple syntax to install any Node.js module −

$ npm install <Module Name>

For example, following is the command to install a famous Node.js web framework module called express −

$ npm install express

Now you can use this module in your js file as following −

var express = require('express');

## **Global vs Local Installation**

By default, NPM installs any dependency in the local mode. Here local mode refers to the package installation in node\_modules directory lying in the folder where Node application is present. Locally deployed packages are accessible via require() method. For example, when we installed express module, it created node\_modules directory in the current directory where it installed the express module.

$ ls -l

total 0

drwxr-xr-x 3 root root 20 Mar 17 02:23 node\_modules

Alternatively, you can use **npm ls** command to list down all the locally installed modules.

Globally installed packages/dependencies are stored in system directory. Such dependencies can be used in CLI (Command Line Interface) function of any node.js but cannot be imported using require() in Node application directly. Now let's try installing the express module using global installation.

$ npm install express -g

This will produce a similar result but the module will be installed globally. Here, the first line shows the module version and the location where it is getting installed.

**express@4.12.2 /usr/lib/node\_modules/express**

├── merge-descriptors@1.0.0

├── utils-merge@1.0.0

├── cookie-signature@1.0.6

├── methods@1.1.1

├── fresh@0.2.4

├── cookie@0.1.2

├── escape-html@1.0.1

├── range-parser@1.0.2

├── content-type@1.0.1

├── finalhandler@0.3.3

├── vary@1.0.0

├── parseurl@1.3.0

├── content-disposition@0.5.0

├── path-to-regexp@0.1.3

├── depd@1.0.0

├── qs@2.3.3

├── on-finished@2.2.0 (ee-first@1.1.0)

├── etag@1.5.1 (crc@3.2.1)

├── debug@2.1.3 (ms@0.7.0)

├── proxy-addr@1.0.7 (forwarded@0.1.0, ipaddr.js@0.1.9)

├── send@0.12.1 (destroy@1.0.3, ms@0.7.0, mime@1.3.4)

├── serve-static@1.9.2 (send@0.12.2)

├── accepts@1.2.5 (negotiator@0.5.1, mime-types@2.0.10)

└── type-is@1.6.1 (media-typer@0.3.0, mime-types@2.0.10)

You can use the following command to check all the modules installed globally −

$ npm ls -g

## **Using package.json**

package.json is present in the root directory of any Node application/module and is used to define the properties of a package. Let's open package.json of express package present in **node\_modules/express/**

{

"name": "express",

"description": "Fast, unopinionated, minimalist web framework",

"version": "4.11.2",

"author": {

"name": "TJ Holowaychuk",

"email": "tj@vision-media.ca"

},

"contributors": [{

"name": "Aaron Heckmann",

"email": "aaron.heckmann+github@gmail.com"

},

{

"name": "Ciaran Jessup",

"email": "ciaranj@gmail.com"

},

{

"name": "Douglas Christopher Wilson",

"email": "doug@somethingdoug.com"

},

{

"name": "Guillermo Rauch",

"email": "rauchg@gmail.com"

},

{

"name": "Jonathan Ong",

"email": "me@jongleberry.com"

},

{

"name": "Roman Shtylman",

"email": "shtylman+expressjs@gmail.com"

},

{

"name": "Young Jae Sim",

"email": "hanul@hanul.me"

} ],

"license": "MIT", "repository": {

"type": "git",

"url": "https://github.com/strongloop/express"

},

"homepage": "https://expressjs.com/", "keywords": [

"express",

"framework",

"sinatra",

"web",

"rest",

"restful",

"router",

"app",

"api"

],

"dependencies": {

"accepts": "~1.2.3",

"content-disposition": "0.5.0",

"cookie-signature": "1.0.5",

"debug": "~2.1.1",

"depd": "~1.0.0",

"escape-html": "1.0.1",

"etag": "~1.5.1",

"finalhandler": "0.3.3",

"fresh": "0.2.4",

"media-typer": "0.3.0",

"methods": "~1.1.1",

"on-finished": "~2.2.0",

"parseurl": "~1.3.0",

"path-to-regexp": "0.1.3",

"proxy-addr": "~1.0.6",

"qs": "2.3.3",

"range-parser": "~1.0.2",

"send": "0.11.1",

"serve-static": "~1.8.1",

"type-is": "~1.5.6",

"vary": "~1.0.0",

"cookie": "0.1.2",

"merge-descriptors": "0.0.2",

"utils-merge": "1.0.0"

},

"devDependencies": {

"after": "0.8.1",

"ejs": "2.1.4",

"istanbul": "0.3.5",

"marked": "0.3.3",

"mocha": "~2.1.0",

"should": "~4.6.2",

"supertest": "~0.15.0",

"hjs": "~0.0.6",

"body-parser": "~1.11.0",

"connect-redis": "~2.2.0",

"cookie-parser": "~1.3.3",

"express-session": "~1.10.2",

"jade": "~1.9.1",

"method-override": "~2.3.1",

"morgan": "~1.5.1",

"multiparty": "~4.1.1",

"vhost": "~3.0.0"

},

"engines": {

"node": ">= 0.10.0"

},

"files": [

"LICENSE",

"History.md",

"Readme.md",

"index.js",

"lib/"

],

"scripts": {

"test": "mocha --require test/support/env

--reporter spec --bail --check-leaks test/ test/acceptance/",

"test-cov": "istanbul cover node\_modules/mocha/bin/\_mocha

-- --require test/support/env --reporter dot --check-leaks test/ test/acceptance/",

"test-tap": "mocha --require test/support/env

--reporter tap --check-leaks test/ test/acceptance/",

"test-travis": "istanbul cover node\_modules/mocha/bin/\_mocha

--report lcovonly -- --require test/support/env

--reporter spec --check-leaks test/ test/acceptance/"

},

"gitHead": "63ab25579bda70b4927a179b580a9c580b6c7ada",

"bugs": {

"url": "https://github.com/strongloop/express/issues"

},

"\_id": "express@4.11.2",

"\_shasum": "8df3d5a9ac848585f00a0777601823faecd3b148",

"\_from": "express@\*",

"\_npmVersion": "1.4.28",

"\_npmUser": {

"name": "dougwilson",

"email": "doug@somethingdoug.com"

},

"maintainers": [{

"name": "tjholowaychuk",

"email": "tj@vision-media.ca"

},

{

"name": "jongleberry",

"email": "jonathanrichardong@gmail.com"

},

{

"name": "shtylman",

"email": "shtylman@gmail.com"

},

{

"name": "dougwilson",

"email": "doug@somethingdoug.com"

},

{

"name": "aredridel",

"email": "aredridel@nbtsc.org"

},

{

"name": "strongloop",

"email": "callback@strongloop.com"

},

{

"name": "rfeng",

"email": "enjoyjava@gmail.com"

}],

"dist": {

"shasum": "8df3d5a9ac848585f00a0777601823faecd3b148",

"tarball": "https://registry.npmjs.org/express/-/express-4.11.2.tgz"

},

"directories": {},

"\_resolved": "https://registry.npmjs.org/express/-/express-4.11.2.tgz",

"readme": "ERROR: No README data found!"

}

## **Attributes of Package.json**

* **name** − name of the package
* **version** − version of the package
* **description** − description of the package
* **homepage** − homepage of the package
* **author** − author of the package
* **contributors** − name of the contributors to the package
* **dependencies** − list of dependencies. NPM automatically installs all the dependencies mentioned here in the node\_module folder of the package.
* **repository** − repository type and URL of the package
* **main** − entry point of the package
* **keywords** − keywords

## **Uninstalling a Module**

Use the following command to uninstall a Node.js module.

$ npm uninstall express

Once NPM uninstalls the package, you can verify it by looking at the content of /node\_modules/ directory or type the following command −

$ npm ls

## **Updating a Module**

Update package.json and change the version of the dependency to be updated and run the following command.

$ npm update express

## **Search a Module**

Search a package name using NPM.

$ npm search express

## **Create a Module**

Creating a module requires package.json to be generated. Let's generate package.json using NPM, which will generate the basic skeleton of the package.json.

$ npm init

This utility will walk you through creating a package.json file.

It only covers the most common items, and tries to guess sane defaults.

See 'npm help json' for definitive documentation on these fields

and exactly what they do.

Use 'npm install <pkg> --save' afterwards to install a package and

save it as a dependency in the package.json file.

Press ^C at any time to quit.

name: (webmaster)

You will need to provide all the required information about your module. You can take help from the above-mentioned package.json file to understand the meanings of various information demanded. Once package.json is generated, use the following command to register yourself with NPM repository site using a valid email address.

$ npm adduser

Username: mcmohd

Password:

Email: (this IS public) mcmohd@gmail.com

It is time now to publish your module −

$ npm publish

If everything is fine with your module, then it will be published in the repository and will be accessible to install using NPM like any other Node.js module.