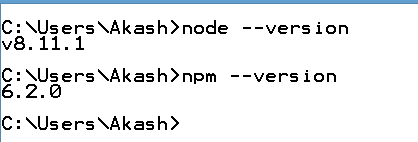
**NPM :** An introduction to Node Package Manager. Today npm is the biggest repository for any programming language, and it has almost every package that you need in a web or mobile development project. Npm means that the node package manager. If you have previous work with any front-end development stuff, then this package manager is familiar to you, In January 2017 over 3,50,000 packages were reported being listed in the npm registry, making it the most significant single language code repository on the planet. At the starting point node package manager is only for Node.js development and nothing else, but as JavaScript community grows, It becomes the backbone of almost all the latest JavaScript framework and Web development. The npm registry hosts the world's most extensive collection of free, reusable code.

Npm is written entirely in JavaScript and was developed by Isaac z. schlueter with inspiration from the shortcoming of other similar projects such as PEAR(PHP) and CPAN(Perl).

You can go to its original website (https://www.npmjs.com/). The Node package manager or npm manages downloads of dependencies of your project.

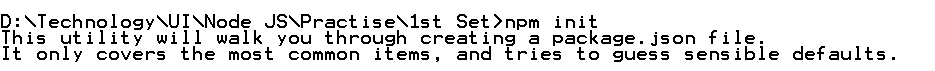
If you have install Node.js in your machine then by default npm has already installed. The node.js comes with npm.



All of your dependencies are written in one file inside your project root called **package.json**

You can create the package.json file using the following command.

**npm init**



Once you run the npm init

it will ask some information from the user.



After given all the required details then



After this we can see there one file with name package.json will create inside the file.

**package.json**

{

"name": "demo-app",

"version": "1.0.0",

"description": "first simple app",

"main": "index.js",

"scripts": {

"test": "jasmine karma"

},

"repository": {

"type": "git",

"url": "http://github.com"

},

"keywords": [

"let",

"const",

"var"

],

"author": "Akash Kale",

"license": "ISC"

}

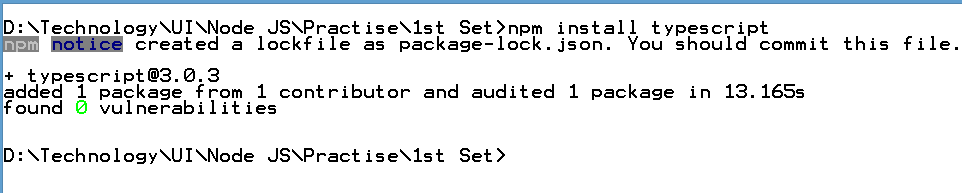
I

If you have already package.json file, but dependencies are not installed previously then you can install all the dependencies defined on your package.json file using following command.

npm install dependenciesname

For example if I want to install the typescript dependencies then we have to use the command as

**npm install typescript**



After execute this command you can see there the one file name with package-lock.json and folder with the name node\_modules will create and what dependencies you install that install inside the node\_modules with that dependencies. The package.json that dependencies information added.

,

"dependencies": {

"typescript": "^3.0.3"

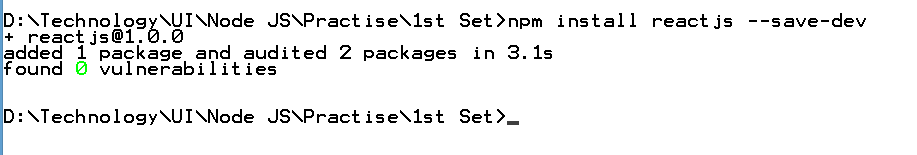
}

If you want to install development, then you can hit the following command with the dev flag.

npm install packageName --save-dev

or

npm install -D packagename



After that the required dependencies added inside the node\_modules and the link for that added in package.json file with the following code.

,

"devDependencies": {

"reactjs": "^1.0.0"

}

The difference between dependencies and devDependencies are usually development tools like a testing library, while dependencies are bundled with the app in production.

If you do not want to write install, then you can use i, and it will do the work for you.

npm i package-name.

If you want to save the dependency inside the package.json file, then you need to add the save flag.

**npm i packagename --save**

**Usage**

npm can manage packages that are local dependencies of a particular project, as well as globally installed JavaScript tools. We can use as a dependency manager for our local project.

**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 contributor to the package

dependencies -> list of the dependencies. npm automatically install 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.

**update npm packages**

you can update npm packages using the following command.

**npm update**

You can also update specific package the following command.

**npm update packageName**

If you are running an old version of npm, then you can update it to the latest release by the following command from root.

**npm install npm -g**

**npm versioning**

In Addition to plain downloads, node package manager manages the versioning so that you can specify any specific version of package.

If you specify the exact version of the npm libraries, then it also help to keep everyone on the same version of library, so that the whole team runs the same version and no conflicts occur until the package.json file is updated.

If you are using the Git version control system, then you need to upload the package.json file and not a node\_modules folder. So when another developer downloads the project, it has already the package.json file, and he only needs to hit the npm install command to up and running with the project.

If we do not specify any version, then it will install latest version of the particular package.