3...

Node Package Manager

Objectives...

- To learn about Node Package Manager.
- To learn how to install packages locally.
- To study how to add dependency in package.json.
- To learn how to install packages globally.
- To learn about updating packages.

3.1 INTRODUCTION

- As we understood in previous chapters that, Node JS is open source server environment which uses JavaScript on the server and available absolutely free.
- You can generate dynamic page contents, can create, open, rewrite, and delete files on the server.
- You can collect form data as well as can add, delete, modify data on database. This all about basic of Node JS to start with NPM (Node Package Manager).
- A package contains all the files needed for a module and modules are the JavaScript libraries that can be included in Node project according to the requirement of the project.

3.2 WHAT IS NPM? [W-22, S-23]

- NPM stands for Node Package Manager. It is written entirely in JavaScript and was developed by Isaac Z. Schlueter in 2010. NPM is the world's largest software registry. Many organizations use npm to manage private development as well.
- It is a package manager for Node JS packages and modules present in an application. So it is basically responsible for managing all your Node JS Packages.

- NPM comes bundled with Node JS. So when you install Node JS, NMP also gets installed and it happens after version 0.6.3 of Node.js. That's why it is a default Package Manager of Node JS.
- NPM is free to use. You can download all npm public software packages without any registration or logon.
- All npm packages are defined in files called **package.json**. The content of Package.jsonmust be written in JSON.
- NPM consists of three distinct components:
  1. **Website:** Use the website to discover packages, set up profiles, and manage other aspects of your npm experience.
  2. **Command Line Interface (CLI):** NPM includes a CLI that can be used to download and install software: C:\>npm install <package>
  3. **Registry:** The registry is a large public database of JavaScript software and the meta-information surrounding it. [S-22]
- An excess of Node JS libraries and applications are published on npm, and many more are added every day. These applications can be searched for on `http://npmjs.org/`. Once you have a package you want to install, it can be installed with a single command line command.
- NPM is very simple to use: you only have to run `npm install async`, and the specified module will be installed in the current directory under ./node_modules/. Once installed to your node_modules folder, you'll be able to use require() on them just like they were built-ins.
- Node Package Manager provides two main functionalities:
  1. It Provides Online repository for Packages/Modules for Node JS which you can easily search online on their official site which is `search.nodejs.org`.
  2. It provides a command line utility to install Node JS Packages. It allows you to do version management and also the dependency management of Node JS packages.

3.2.1 Need for NPM [S-22]

1. It helps in incorporating the pre-built packages into our project.
2. It assists in downloading various standalone tools which can be used right away.
3. Using NPM, you can even run packages without having to download it.
4. Developers often use NPM to share their code with other NPM users.
5. Helps in restricting the code to specific developers and forming virtual teams using orgs.

6. Helps in managing and maintaining various versions of codes and their dependencies.
7. NPM also updates the application with the update in underlying codes.

3.2.2 Advantages of NPM

1. NPM is free to use. As it is open source package system.
2. It is simpler than SOAP.
3. It is a default package manager for Node JS.
4. Completely open source and written in JavaScript
5. As it manages all the packages and modules for Node JS and it consists of also command line client NPM, so it's become world's largest software registry.

3.3 PACKAGE [S-22, W-22, S-23]

- Package contains all the files which are needed for modules and modules are the JavaScript libraries that can be included in a Node JS project as per the requirements of the project.
- The npm registry contains packages, A package is a file or directory that is described by a Package.json file. A package must contain a Package.json file in order to be published to the npm registry.

What is the Package.json file?

- The Package.json file is one of the most important files when working with Node applications. With this single file, we keep track of all of our external dependencies. This file is also used to tell others about our project. The "package.json" file is used to hold the metadata about a particular project. This information provides the node package manager the necessary information to understand how the project should be handled along with its dependencies.
- The Package.json file contains information such as the project description, the version of the project in a particular distribution, license information, and configuration data.
- The Package.json file is usually located at the root directory of a Node JS project.
- Let's take an example of how the structure of a module looks when it is installed via npm. The below snapshot shows the file contents of the express module when it is included in your Node JS project. From the snapshot, you can see the Package.json file in the express folder.

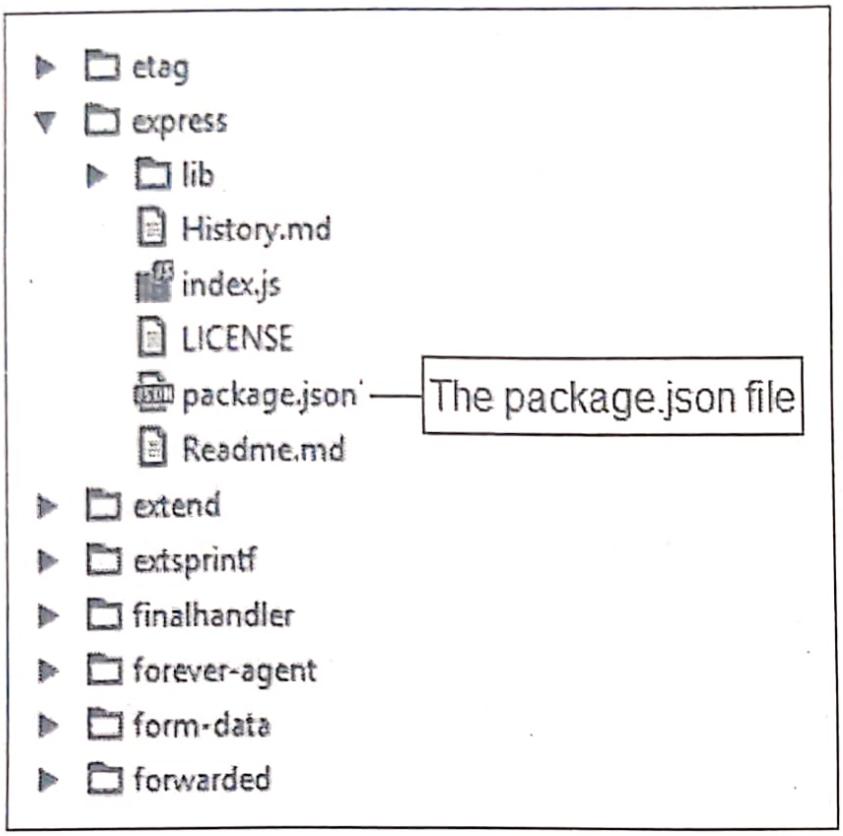

Fig. 3.1

- If you open the Package.json file you will see a lot of information in the file.
- See following figure, the express@~4.13.1 mentions the version number of the express module being used.

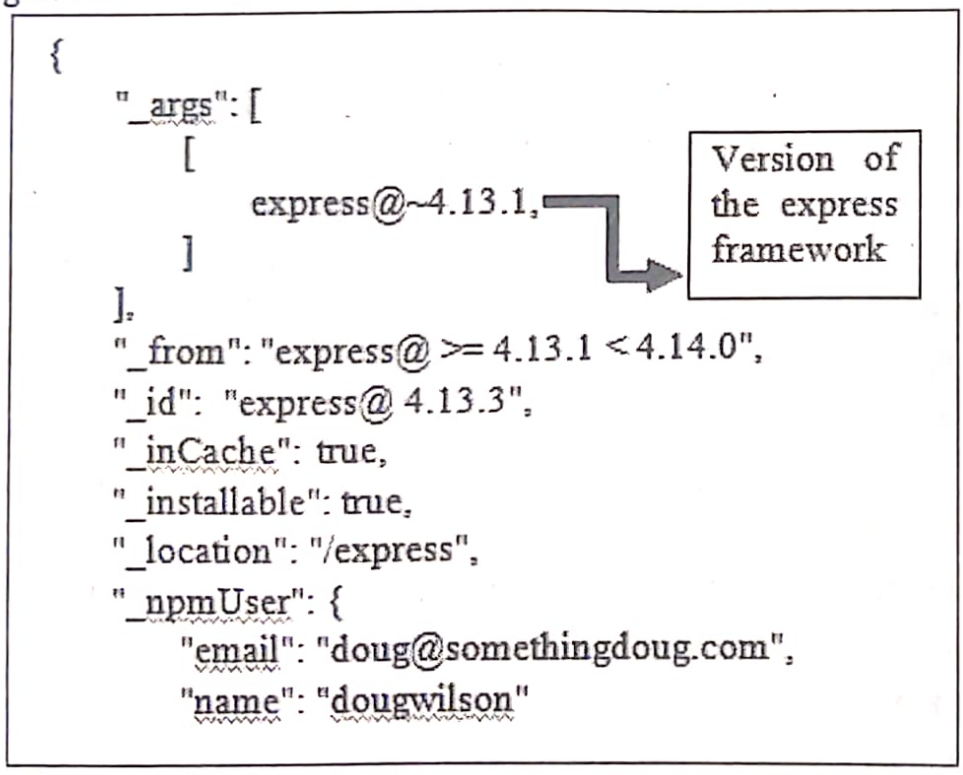

Fig. 3.2

3.4 INSTALLING PACKAGES LOCALLY [W-22]

- Now, we will look how to install packages. There are two ways of installation the packages, either locally or globally. So the question arises when should we install globally and when should we install locally. You chose the method which based on how actually you want to use those packages.
- You should install packages locally when you want to depend on the package from your own module, using something such as Node JS require. This is npm install's default behavior.
- When you want to use a package as a command line tool, (like grunt CLI), then you have to install it globally.

3.4.1 Installing a Package

- You can download a package with the command: `npm install <package_name>`

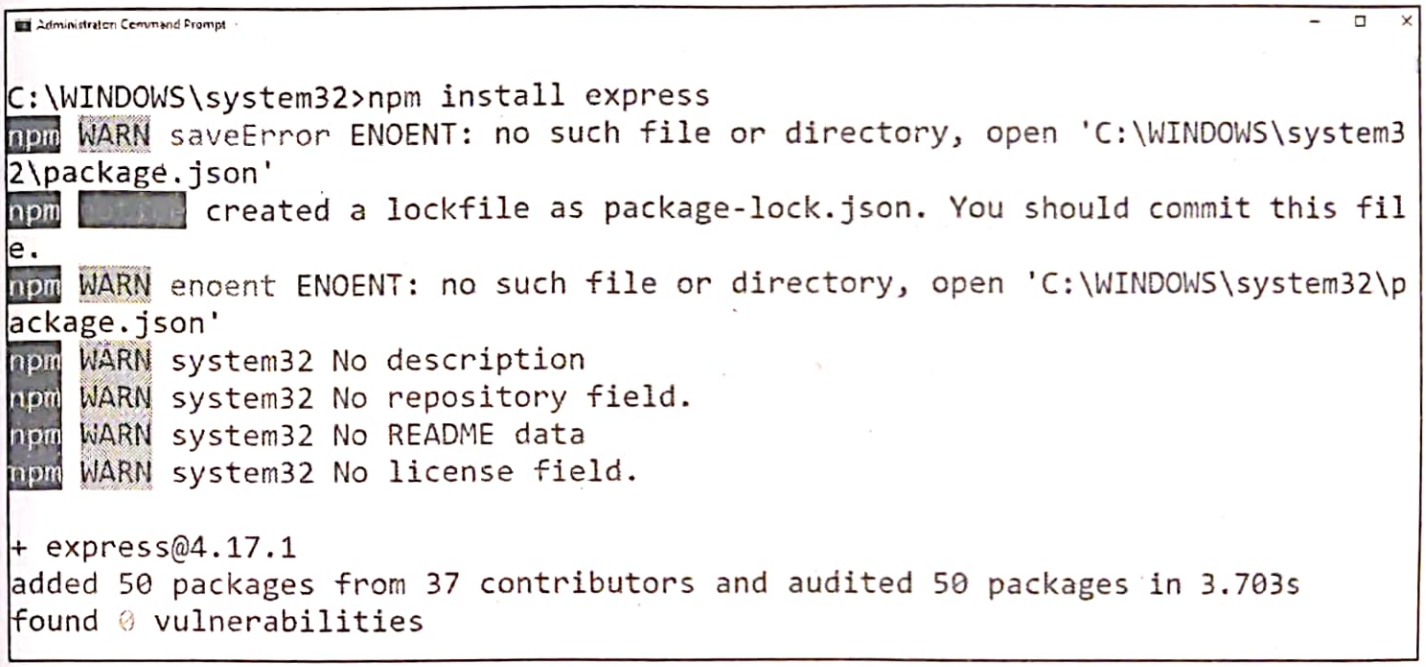

Fig. 3.3

- This creates the node_modules directory in your current directory (if one doesn't exist yet) and downloads the package to that directory.
- If you want to confirm that npm install worked correctly, you should check to see that a node_modules directory exists and that it has a directory for the package(s) that you have installed.
- For example, Take for instance that you install a package called express, you can confirm that it worked correctly by checking that a node_modules directory now exists and that the directory has a subdirectory named express.

For Microsoft Windows:

```
npm install express
C:\ dir node_modules
#=> express
```

- If a Package.json file does not exist in the local directory, the latest version of the package will be installed.
- However, if a Package.json file exists, npm will install the latest version that satisfies the semver (semantic versioning) rule that is declared in the package.json.

3.4.2 Using the Installed Package in Your Code

- Once you have run `npm install <package name>` and the package is in the node_modules directory, it can be used in your code. For instance, when you are creating a Node JS module, you can use require to access it.
- **Example:** Create a file and name it index.js, and add the following code:

```
// index.js
var express = require('express');
app.use(express)
console.log("successfully required a package")
```

When you run the code, you should get the output: `successfully required a package.`

- If you had not installed express properly, you will get the following error message:

```
module.js:340
throw err;
      ^
Error: Cannot find module 'express'
```

- To fix this error, you should run npm install express in the same directory as your index.js

3.4.3 How to Uninstall Local Packages

- If you want to remove a package directly from your node_modules directory, you should use:

```
npm uninstall <package>
npm uninstall lodash
```

- If you want to remove it from the dependencies in package.json, you have to use the save flag:

```
npm uninstall --save lodash
```

Note: if you had installed the package as a "devDependency" (i.e. with `--save-dev`) then `--save` will not remove it from package.json. You will need to use `--save-dev` to uninstall it.

- If you want to confirm that npm uninstall worked correctly, you should find the node_modules directory. Ensure that it no longer contains a directory for the package(s) you uninstalled.
- This can be done by running:

  `dir node_modules on Windows.`
- **For example,** Install a package called axios. Then confirm that it run successfully by listing the contents of the node_modules directory and seeing a directory called axios.
- Uninstall axios with npm uninstall. Then confirm that it run successfully by listing the contents of the node_modules directory and confirming the absence of a directory called axios.

Install Axios:

```
npm install axios
dir node_modules            # use `ls node_modules` for Unix
```

Uninstall Axios:

```
#=> axios
> npm uninstall axios
> dir node_modules          # use `ls node_modules` for Unix
#=>
```

3.5 ADDING DEPENDENCY IN PACKAGE.JSON [W-22]

- We can add dependencies to a `Package.json` in two ways:
  1. File from the Command Line.
  2. Manually Editing the Package.json File.

Adding Dependencies to a `Package.json` File from the Command Line:

- To add dependencies and devDependencies to a `Package.json` file from the command line, we can install them in the root directory of package using the `--save-prod` flag for dependencies (the default behavior of npm install) or the `--save-dev` flag for devDependencies.
- To add an entry to the `"dependencies"` attribute of a `Package.json` file, on the command line, following command will be used:

  `npm install <package-name> [--save-prod]`
- To add an entry to the `"devDependencies"` attribute of a `Package.json` file, on the command line, run the following command:

  `npm install <package-name> --save-dev`

Manually Editing the `Package.json` File:

- To add dependencies to a Package.json file, in a text editor, add an attribute called "dependencies" that references the name and semantic version of each dependency:

```
{
  "name": "my_package",
  "version": "1.0.0",
  "dependencies": {
    "my_dep": "^1.0.0",
    "another_dep": "~2.2.0"
  }
}
```

- To add devDependencies to a `Package.json` file, in a text editor, add an attribute called "devDependencies" that references the name and semantic version of each devDependency:

```
"name": "my_package",
"version": "1.0.0",
"dependencies": {
  "my_dep": "^1.0.0",
  "another_dep": "~2.2.0"
},
"devDependencies" : {
  "my_test_framework": "^3.1.0".
  "another_dev_dep": "1.0.0 - 1.2.0"
}
```

3.6 INSTALLING PACKAGES GLOBALLY [S-22]

- In previous section, we talked about working with Package.json, Now, will look on how to install, update and uninstall packages globally. As we have discussed earlier that, there are two options available to install a package: it is either install it locally or install it globally. The choice on which kind of installation is dependent on how you want to use the package.
- Whenever you want to use a package as a command line tool, you should install it globally. In this way, it will work no matter which directory is current. This is the choice you should use if you were installing grunt.

- Whereas, when you want to depend on the package from your own module, you should install it locally. This is the choice you would normally use if you are using require statements.
- You can use the command npm install -g <package>, for install:

```
npm install -g <package_name>
```

3.6.1 How to Uninstall Global Packages

- For you to uninstall a package all you need to do is to type:

```
npm uninstall -g <package>
```

- If you want to uninstall a package called jshint, you would type:

```
npm uninstall -g jshint
```

3.7 UPDATING PACKAGES

Updating Local Packages:

- You should periodically update the packages that your application depends on. Then if there are code changes made by the original developers, your code will also be improved.
- To do this:

1. Navigate to the root directory of your project and ensure it contains a Package.json file:

```
cd /path/to/project
```

2. In your project root directory, run the update command:

```
npm update
```

3. To test the update, run the outdated command. There should not be any output.

```
npm outdated
```

Updating Global Packages

- This requires version 2.6.1 or greater.
- If you want to update packages, you should type this command on your terminal:

```
npm update -g <package>
```

- For instance, if you want to update a package called grunt, you would type:

```
npm update -g grunt
```

- If you want to find out the packages that needs to be updated, type:

```
npm outdated -g --depth=0
```

- Finally, if you want to update all global packages, you should type:

```
npm update -g
```

- For any npm version that is below 2.6.1, you should run this script:

```
#!/bin/sh
set -e
set -x
for package in $(npm -g outdated --parseable --depth=0 | cut -d: -f3)
do
npm -g install "$package"
done
```

To update all outdated global packages.

- However, it is recommended that you upgrade to the latest version of npm. You can do this by typing:

```
npm install npm@latest -g
```

3.8 MANAGING THIRD PARTY PACKAGES WITH NPM [W-22, S-23]

- As we have seen, the "Node package manager" has the ability to manage modules, which are required by Node JS applications.
- Let's look at some of the functions available in the node package manager for managing modules.

1. **Installing packages in global mode:** Modules can be installed at the global level, which just basically means that these modules would be available for all Node JS projects on a local machine. The below example shows, how to install the "express module" with the global option.

```
npm install express -global
```

The global option in the above statement is what allows the modules to be installed at a global level.

2. **Listing all of the global packages installed on a local machine:** This can be done by executing the below command in the command prompt:

```
npm list --global
```

Below is the output which will be shown, if you have previously installed the "express module" on your system. Here you can see the different modules installed on the local machine.

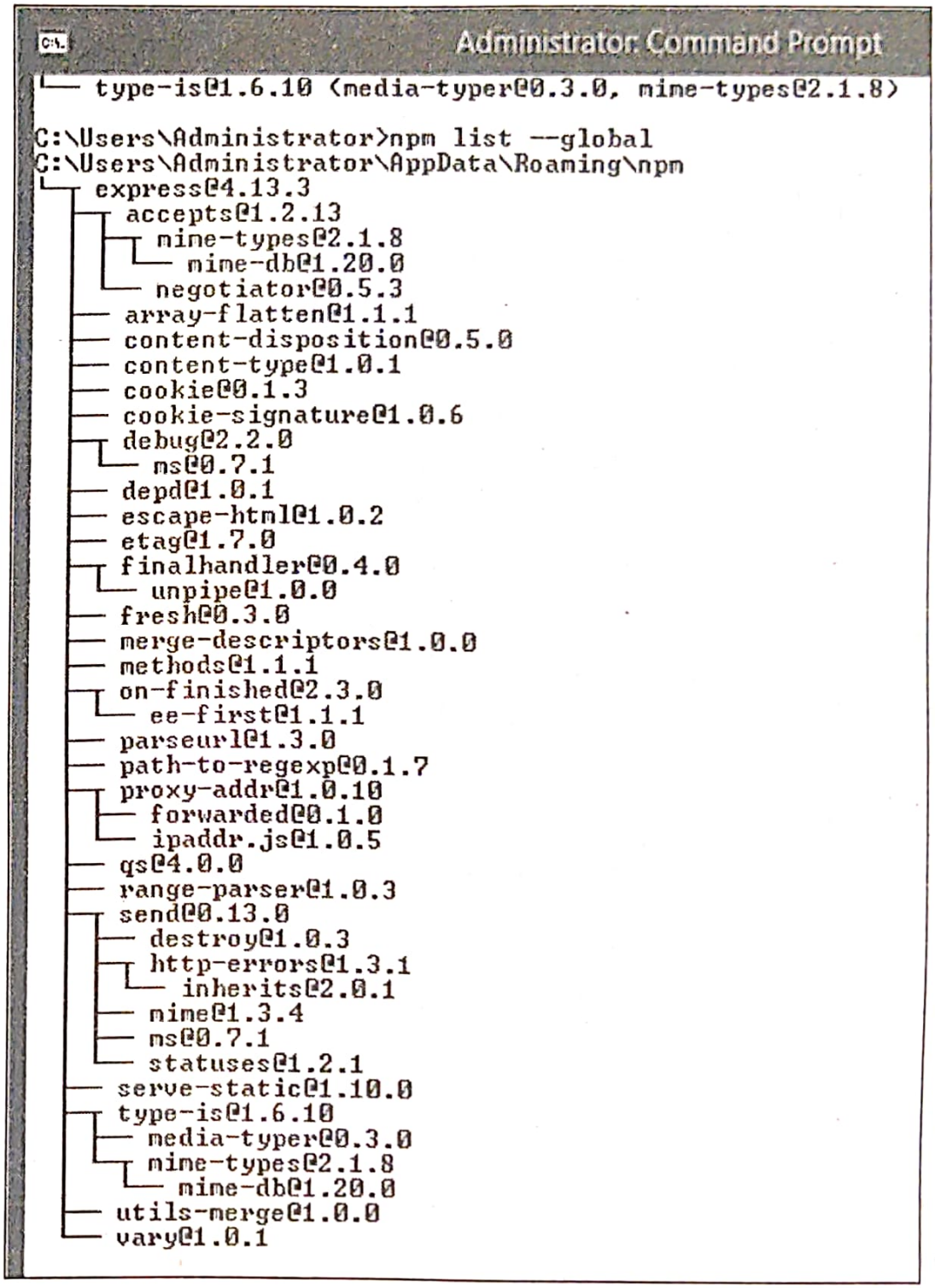

Fig. 3.4

3. **Installing a specific version of a package:** Sometimes there may be a requirement to install just the specific version of a package. Once you know what are the package and the relevant version that needs to be installed, you can use the npm install command to install that specific version. The example below shows how to install the module called underscore with a specific version of 1.7.0.

```
npm install underscore@1.7.0
```

- To install a package globally, add an extra -g tag in syntax like npm install package_name -g.
- To install a package and simultaneously save it in package.json, add -save flag. The -save flag is default in npm install command so it is equal to npm install package_name command.
- To uninstall packages using npm, use the syntax: npm uninstall
- To uninstall global packages, follow the syntax: npm uninstall package_name -g

Check Your Understanding:

1. Node JS is written in which language_____.
   (a) JavaScript (b) C
   (c) C++ (d) All of the above
2. Which function is used to include modules in Node JS?
   (a) include( ); (b) require( );
   (c) attach( ); (d) None of the above
3. Which statement executes the code of sample.js file?
   (a) nodejs sample.js (b) node sample.js
   (c) sample.js (d) None of the above
4. The $ npm ls -g statement is used to list down all the locally installed module?
   (a) True (b) False
5. How can we check the current version of NPM?
   (a) npm -version (b) npm --ver
   (c) npm help (d) None of the above
6. To install Node JS express module_____is used
   (a) $ npm install express (b) $ node install express
   (c) $ install express (d) None of above
7. What is the default scope in Node js application?
   (a) Global (b) Local
   (c) Public (d) Private
8. For what npm stands?
   (a) Node Project Manager (b) Node Package Manager
   (c) New Project Manager (d) New Package Manager
9. Command to list all modules that are install globally?
   (a) $ npm ls -g (b) $ npm ls
   (c) $ node ls -g (d) $ node ls

4. **Updating a package version:** Sometimes you may have an older version of a package in a system, and you may want to update to the latest one available in the market. To do this one can use the npm update command. The example below shows how to update the underscore package to the latest version.

```
npm update underscore
```

5. **Searching for a particular package:** To search whether a particular version is available on the local system or not, you can use the search command of npm. The example below will check if the express module is installed on the local machine or not.

```
npm search express
```

6. **Uninstalling a package:** The same in which you can install a package, you can also uninstall a package. The uninstallation of a package is done with the uninstallation command of npm. The example below shows how to uninstall the express module.

```
npm uninstall express
```

Summary

- NPM stands for Node Package Manager and it works as a project manager for JavaScript.
- NPM (Node Package Manager) is the default package manager for Node.js and is written entirely in JavaScript. Developed by Isaac Z.
- A package contains all the files needed for a module and modules are the JavaScript libraries that can be included in Node project according to the requirement of the project.
- NPM can install all the dependencies of a project through the package.json file. It can also update and uninstall packages.
- The Node Package Manager (npm) is used to download and install modules which can then be used in a Node JS application.
- The Node package manager has a complete set of commands to manage the npm modules on the local system such as the installation, un-installation, searching, etc
- Node Package Manager (NPM) provides two main functionalities:
  1. Online repositories for node.js packages/modules which are searchable on search.nodejs.org.
  2. Command line utility to install Node.js packages, do version management and dependency management of Node.js packages.
- Version of npm installed on system can be checked using `npm -v`.
- To install packages and modules in the project use the `npm install package_name` syntax.

- To install a package globally, add an extra -g tag in syntax like `npm install package_name -g`.
- To install a package and simultaneously save it in `package.json`, add `-save` flag. The –save flag is default in npm install command so it is equal to `npm install package_name` command.
- To uninstall packages using npm, use the syntax: `npm uninstall`
- To uninstall global packages, follow the syntax: `npm uninstall package_name -g`

Check Your Understanding

1. Node JS is written in which language_____.
   (a) JavaScript (b) C
   (c) C++ (d) All of the above
2. Which function is used to include modules in Node JS?
   (a) include( ); (b) require( );
   (c) attach( ); (d) None of the above
3. Which statement executes the code of sample.js file?
   (a) nodejs sample.js (b) node sample.js
   (c) sample.js (d) None of the above
4. The $ npm ls -g statement is used to list down all the locally installed module?
   (a) True (b) False
5. How can we check the current version of NPM?
   (a) npm –version (b) npm --ver
   (c) npm help (d) None of the above
6. To install Node JS express module_____is used.
   (a) $ npm install express (b) $ node install express
   (c) $ install express (d) None of above
7. What is the default scope in Node.js application?
   (a) Global (b) Local
   (c) Public (d) Private
8. For what npm stands?
   (a) Node Project Manager (b) Node Package Manager
   (c) New Project Manager (d) New Package Manager
9. Command to list all modules that are install globally?
   (a) $ npm ls -g (b) $ npm ls
   (c) $ node ls -g (d) $ node ls

10. Which of the following is true about package.json?
    (a) package.json is present in the root directory of any Node application/module.
    (b) package.json is used to define the properties of a package.
    (c) package.json can be used to update dependencies of a Node application.
    (d) All of the above.

ANSWER KEY

1. (d)	2. (b)	3. (b)	4. (b)	5. (a)
6. (a)	7. (b)	8. (b)	9.(a)	10. (d)

Practice Questions

Q.I: Answer the following questions in short.

1. Write syntax to install local package?
2. Write syntax to install global package?
3. How can you check the installed version of Node JS?
4. For what require( ) is used in Node JS?
5. List out NPM components.
6. What is use of Registry?

Q.II: Answer the following questions.

1. What is npm? Explain with its advantages.
2. What is the Package.json file? What is it used for?
3. Define the functionalities of NPM?
4. Explain some popular modules in Node JS?
5. How to add dependency into package.json?

Q.III: Define the terms.

1. NPM
2. Package
3. Json Package
4. Local Package
5. Global package

❖❖❖