Node Package Manager

Akash Pundir

Node Package Manager, commonly known as npm, is the default package manager for Node.js. It is **a command-line tool that allows developers to easily manage and install JavaScript libraries and tools for use in their projects**. npm simplifies the process of adding, updating, and removing dependencies in a Node.js project.

package.json

The package.json file is a metadata file commonly used in Node.js projects to manage project configuration, dependencies, and other settings. **It provides information about the project, its dependencies, scripts for running various tasks, and more.**

Let’s make our first node application

- Open Command Prompt, make a new folder, and run this command.

npm init

This command will start the initialization process and prompt you with a series of questions.

- **name**: The name of your project. It should be lowercase and may include letters, numbers, hyphens, and underscores.
- **version**: The initial version of your project. Follows the Semantic Versioning (SemVer) format (major.minor.patch).
- **description**: A brief description of your project (optional).
- **entry point**: The main entry point file for your application (default is index.js).
- **test command**: Command to run your tests (optional).
- **git repository**: The URL of your project's Git repository (optional).
- **keywords**: Keywords to help others discover your project on npm (optional).
- **author:** Your name (optional).
- **license:** The license under which your project is distributed (default is ISC).
- Is this OK? (yes): Confirm that the information is correct.

package.json vs package-lock.json

- The package.json file focuses on project metadata and specifying the desired versions of dependencies, while the package-lock.json file ensures deterministic installations by locking the exact versions of dependencies and their dependencies.

Make a new index.js file in same directory

```
const http = require('http');

const server = http.createServer((req, res) => {
  res.end('Hello, this is your Node.js server!');
});

const port = 3000;

server.listen(port, () => console.log(`Server is running
on http://localhost:${port}`));
```

Now type command

node index.js

Custom NPM Modules

Akash Pundir

What is a package?

A "package" typically refers to a collection of code files, assets, and configuration files bundled together for a specific purpose. Packages are used to organize and distribute code in a modular and reusable way.

Create a package.json file

- To create a package.json file, on the command line, in the root directory of your Node.js module :

npm init

Let’s try a package

npm install lodash

Now let’s use lodash

```
// Create app.js file
const q = require('lodash');

const numbers = [1, 2, 3, 4, 5];

const doubledNumbers = q.map(numbers, n => n * 2);

console.log('Original numbers:', numbers);
console.log('Doubled numbers:', doubledNumbers);
```

Let’s make our own package module and use it

- Make a new **add.js** file in the same directory

```
const add= (a,b)=> a+b;

module.exports={
    add
}
```

Now, go back to app.js file and import your module

```
const a=require('./add.js')

let p=a.add(2,4);
console.log(p);
```

Provide responses for the required fields (name and version), as well as the main field:

- name: The name of your module.
- version: The initial module version. NPM recommend following semantic versioning guidelines and starting with 1.0.0.

Create the file that will be loaded when your module is required by another application

- In the file, add a function as a property of the exports object. This will make the function available to other code:

```
module.exports = function insertSpace(inputString) {
    if (typeof inputString !== 'string') {
      throw new Error('Input must be a string');
    }

    // Use regular expression to insert space between each letter
    const spacedString = inputString.replace(/./g, '$& ');

    // Remove the extra space at the end
    return spacedString.trim();
  };
```

Publish your package to npm:

npm publish

Make sure you are logged in to your npm accounts (npm adduser) before using this command.