NPM - Intro

dependency manager

Assembler
School of Software Engineering

Assembler
School of Software Engineering

Index

Before using a dependency manager

Until now you have had to import various dependencies to your projects like **jQuery**, **Bootstrap** ... But manually (downloading and importing them to your project) or by CDN.

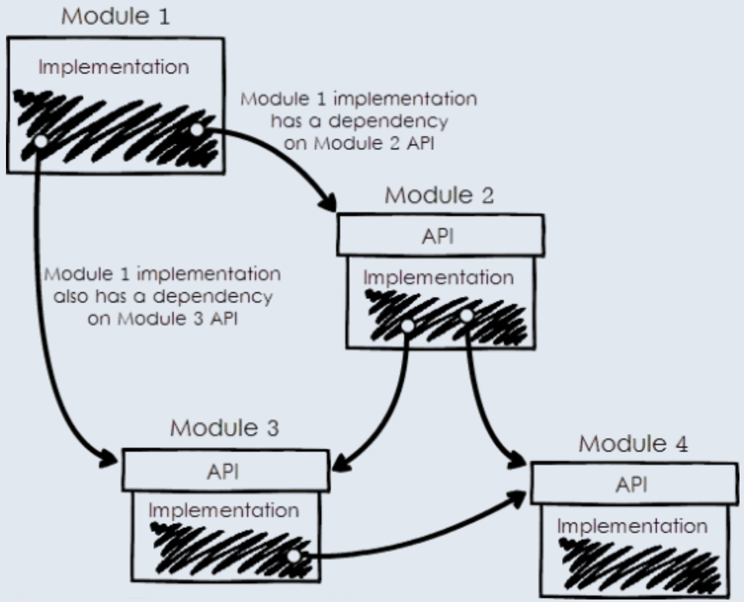
In the case of a dependency downloaded in your project, if you want to **update** the version of the dependency you will have to **download and import it manually**.

In the case of using CDN, as the **dependency is not downloaded**, you depend on making a **request**.

Dependencies of dependencies

In addition, there are **dependencies that may depend on other dependencies** and you do not have to know it.

Dependency managers also take care of this, thus **avoiding conflicts between versions**.


Module 1
Implementation
Module 1 implementation has a dependency on Module 2 API
Module 2
API
Implementation
Module 1 implementation also has a dependency on Module 3 API
Module 3
API
Implementation
Module 4
API
Implementation

Assembler
School of Software Engineering

What is a dependency manager?

A **dependency manager** is a tool that **coordinates** and **centralizes** the integration of libraries or external packages to use them in your project. In this way you have a tool focused only in this specific task and facilitates the sustainability of your project by providing facilities to **import new dependencies** or **update current ones**.

What is a NPM?

NPM is a tool that **manages the dependencies** of our **project, both in client and in backend**.

Thanks to this tool, we can work with the **installation of those dependencies that our project needs in a comfortable way**, avoiding all the problems involved in installing the dependencies manually and without any control or supervision.

What is NodeJS?

NodeJS is an open source, cross-platform, **runtime environment** for the **server layer** (but not limited to it) based on the **JavaScript** programming language.

NPM is the default **dependency manager** of NodeJS, therefore to be able to use NPM we simply have to install NodeJS on our computer.

Assembler
School of Software Engineering

What is “package.json” in NPM?

When you **initialize NPM** in your project, “package.json” file will be created.

This file contains all the main information about the project as for example:

- Project name
- Version
- Description
- Dependencies
- Scripts
- Repository
- etc

```
{
  "name": "myPackage",
  "version": "2.0.0",
  "description": "Super project!",
  "scripts": {
    "test": "jest --verbose"
  },
  "author": "Assembler School",
  "license": "ISC",
  "dependencies": {
    "ejs": "^2.6.1",
    "express": "^4.16.4"
  },
  "devDependencies": {
    "jest": "^24.8.0"
  }
}
```

Assembler
School of Software Engineering

What is “package-lock.json” in NPM?

The **package-lock.json is automatically generated for any operations where npm modifies either the node_modules tree, or package.json**.

It describes the exact tree that was generated, such that subsequent installs are able to generate identical trees, regardless of intermediate dependency updates.

```
{
  "name": "yuust",
  "version": "2.0.0",
  "lockfileVersion": 1,
  "requires": true,
  "dependencies": {
    "@babel/code-frame": {
      "version": "7.0.0",
      "resolved": "https://registry.npmjs.org/@babel/code-frame/-/code-frame-7.0.0.tgz",
      "integrity": "sha512-OfC2uemaknXr87bdLUkWog7nYuliM9Ij5HUcajsVcMCpQrcLmtxRbVFT",
      "requires": {
        "@babel/highlight": "^7.0.0"
      }
    },
    "@babel/core": {
      "version": "7.4.5",
      "resolved": "https://registry.npmjs.org/@babel/core/-/core-7.4.5.tgz",
      "integrity":
"sha512-OvjIh6aqXtlsA8ujtGKfC7LYWksYSX8yQcM8Ay3LuvVeQ63lcOKgoZWVqcpFwkd29aYU9r
Vx7jxhfhiEDV9MZA==",
      "requires": {
…..
```

package-lock.json features

- **Describe a single representation of a dependency tree** such that teammates, deployments, and continuous integration are guaranteed to install exactly the same dependencies.

- Provide a facility for users to "**time-travel" to previous states of node_modules** without having to commit the directory itself.

- To facilitate **greater visibility of tree changes** through readable source control diffs.

- And **optimize the installation process by allowing npm to skip repeated metadata resolutions for previously-installed packages.**

Assembler
School of Software Engineering

Types of dependencies

In NPM we can find different types of dependencies, some of the most common are:

Production: These are mandatory dependencies for running the project. Production dependencies are modules which are also required at runtime.

Development: These are dependencies which are needed at the time of development but are not responsible for working of the application i.e. even if we skip these dependency our application will work just fine. Ex: jest

Global: Global dependencies are those that your operating system needs to work on a regular basis. For example nodemon. In many of your projects you can use this external tool to work without it implying that the project needs it as a dependency

i Different types of dependencies

Assembler
School of Software Engineering

Advantages of using NPM

- Quickly and easily import the dependencies you want to include in your project
- Centralize dependency management in a single tool
- Keeping your dependencies updated with latest patch, release or major version very easy
- Separate the dependencies depending on the use and importance that will be given in the project

Alternatives to NPM

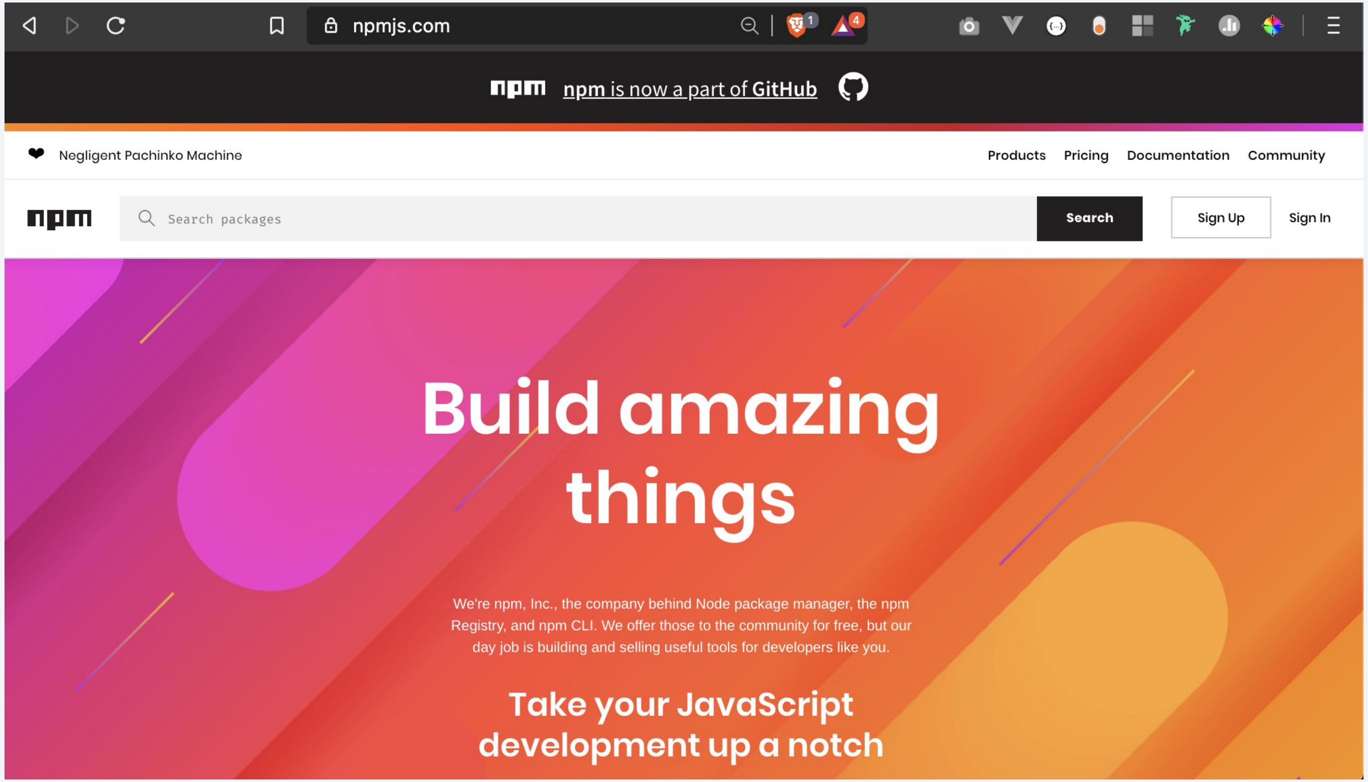
YARN is a **JavaScript dependency manager**, which is focused on speed and security, and unlike other managers like NPM, YARN is **very fast and very easy to use**.

YARN was born at a time when NPM was very stagnant, since the community requested a series of improvements that did not come as quickly as required, and YARN appears in that context.

Currently NPM and YARN are very similar tools, because thanks to the push that YARN has been exerting when applying new functionalities, it has served so that NPM has also been improving at the same time

Assembler
School of Software Engineering

Find Public packages


npmjs.com

npm is now a part of GitHub

Negligent Pachinko Machine

Products Pricing Documentation Community

Search packages

Search

Sign Up

Sign In

Build amazing things

We're npm, Inc., the company behind Node package manager, the npm Registry, and npm CLI. We offer those to the community for free, but our day job is building and selling useful tools for developers like you.

Take your JavaScript development up a notch

Assembler
School of Software Engineering

Official Documentation

https://docs.npmjs.com/cli/npm

Assembler
School of Software Engineering

Questions?