

# Managing Packages with npm



**Kamran Ayub**

Helping developers get started with JavaScript

[www.linkedin.com/in/kamranayub](https://www.linkedin.com/in/kamranayub)

# Referencing Third-party Scripts

For browser-based applications, you can reference third-party scripts

```
<script type="text/javascript" src="lib/jquery.min.js"></script>
<script type="text/javascript">
(function () {
    const container = $('.container');
})();
</script>
```



# Referencing Third-party Packages

For module-based applications, you reference npm packages

```
const $ = require('jquery');

const container = $('.container');
```



# B-Roll: npmjs.com

 Pro Teams Pricing Documentation



Search packages

Search

Sign Up

Sign In

# Build amazing things

We're GitHub, the company behind 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

```
npm install
```

◀ Installs project dependencies

```
npm install <package>
```

◀ Installs specific package

```
npm update <package>
```

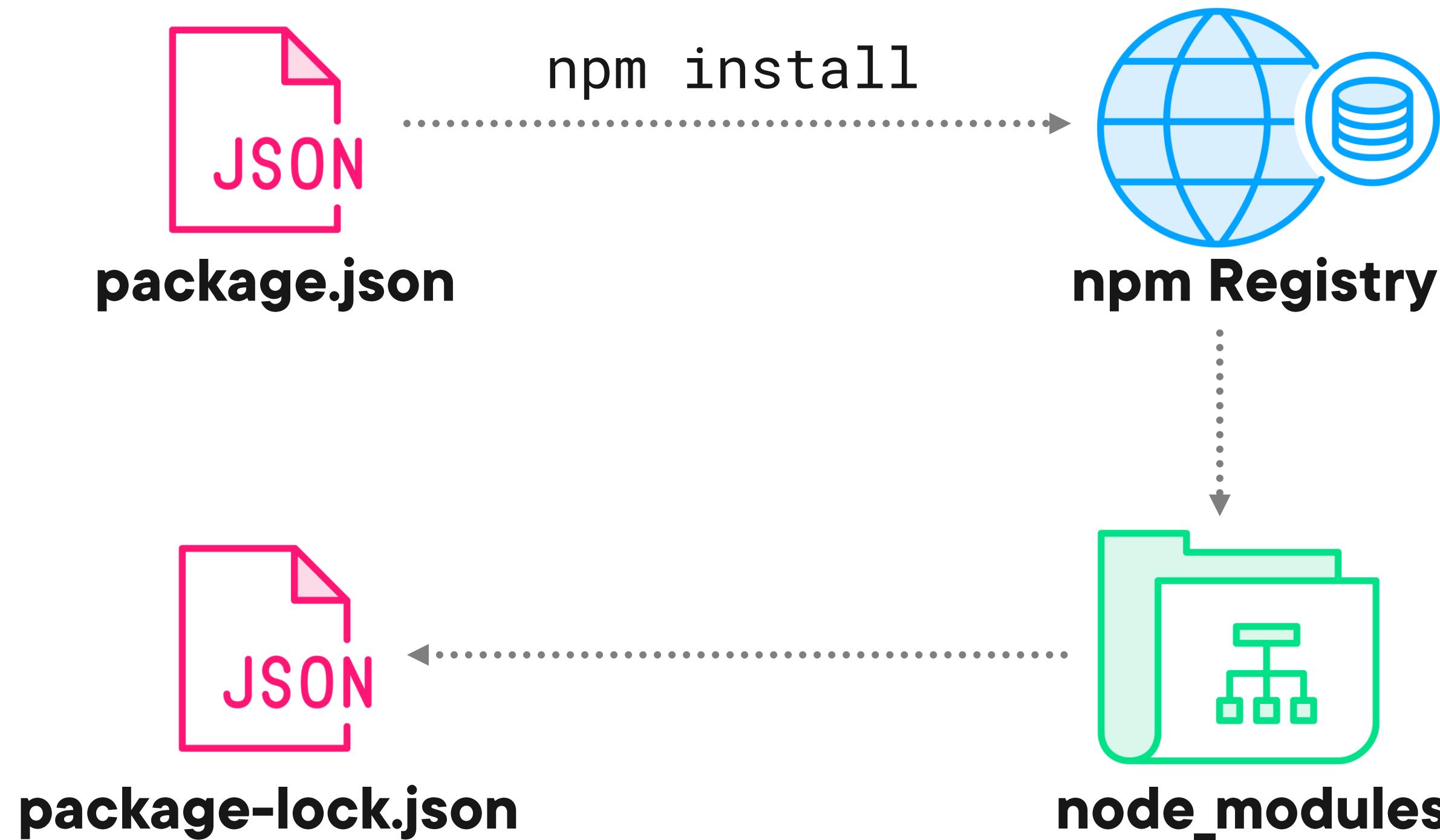
◀ Update a package

```
npm uninstall <package>
```

◀ Removes a package



# How npm Works



# The Heart of a JavaScript Project

The **package.json** file contains metadata about the project, including dependencies

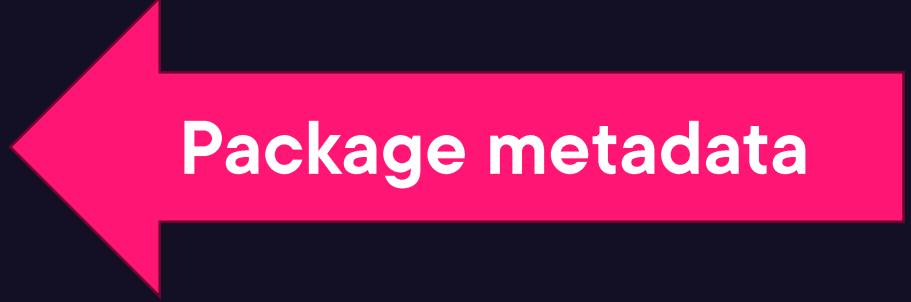
```
{  
  "name": "package-name",  
  "version": "1.0.0",  
  "dependencies": {  
    "express": "5.x"  
  },  
  "devDependencies": {  
    "eslint": "^8.0.0"  
  }  
}
```



# The Heart of a JavaScript Project

The **package.json** file contains metadata about the project, including dependencies

```
{  
  "name": "package-name",  
  "version": "1.0.0",  
  "dependencies": {  
    "express": "5.x"  
  },  
  "devDependencies": {  
    "eslint": "^8.0.0"  
  }  
}
```



Package metadata



# The Heart of a JavaScript Project

The **package.json** file contains metadata about the project, including dependencies

```
{  
  "name": "package-name",  
  "version": "1.0.0",  
  "dependencies": {  
    "express": "5.x"  
  },  
  "devDependencies": {  
    "eslint": "^8.0.0"  
  }  
}
```



A diagram illustrating the structure of a package.json file. On the left, the JSON code is shown. Two arrows point from specific keys to text boxes: a pink arrow points from "dependencies" to a pink box labeled "Runtime dependencies"; a yellow arrow points from "devDependencies" to a yellow box labeled "Development-time only dependencies".

**Runtime dependencies**

**Development-time only dependencies**



# The Heart of a JavaScript Project

The **package.json** file contains metadata about the project, including dependencies

```
{  
  "name": "package-name",  
  "version": "1.0.0",  
  "dependencies": {  
    "express": "5.x"  
  },  
  "devDependencies": {  
    "eslint": "^8.0.0"  
  }  
}
```

Package ID



# The Heart of a JavaScript Project

The **package.json** file contains metadata about the project, including dependencies

```
{  
  "name": "package-name",  
  "version": "1.0.0",  
  "dependencies": {  
    "express": "5.x"  
  },  
  "devDependencies": {  
    "eslint": "^8.0.0"  
  }  
}
```

Package ID

Version Constraint



# Package Version Matching

A version string provides an expression to match against

package.json

```
{  
  "name": "package-name",  
  "version": "1.0.0",  
  "dependencies": {  
    "express": "5.x"  
  }  
}
```



# Package Version Matching

A version string provides an expression to match against

package.json

```
{  
  "name": "package-name",  
  "version": "1.0.0",  
  "dependencies": {  
    "express": "5.x"  
  }  
}
```

```
$ npm view express versions  
[  
  '0.14.0',  
  '1.0.0',  
  '4.5.1',  
  '5.1.1',  
  '5.2.6',  
  '5.6.0',  
]
```



# Package Version Matching

A version string provides an expression to match against

package.json

```
{  
  "name": "package-name",  
  "version": "1.0.0",  
  "dependencies": {  
    "express": "5.x"  
  }  
}
```

```
$ npm view express versions  
[  
  '0.14.0',  
  '1.0.0',  
  '4.5.1',  
  '5.1.1',  
  '5.2.6',  
  '5.6.0',  
]
```



# Package Version Matching

A version string provides an expression to match against

package.json

```
{  
  "name": "package-name",  
  "version": "1.0.0",  
  "dependencies": {  
    "express": "5.x"  
  }  
}
```

Latest matching version

```
$ npm view express versions  
[  
  '0.14.0',  
  '1.0.0',  
  '4.5.1',  
  '5.1.1',  
  '5.2.6',  
  '5.6.0',  
]
```



# Package Version Matching

A version string provides an expression to match against

package.json

```
{  
  "name": "package-name",  
  "version": "1.0.0",  
  "dependencies": {  
    "express": "5.x"  
  }  
}
```

Newly published

```
$ npm view express versions  
[  
  '0.14.0',  
  '1.0.0',  
  '4.5.1',  
  '5.1.1',  
  '5.2.6',  
  '5.6.0',  
  '5.7.0',  
]
```



# The Nervous System of a JavaScript Project

The **package-lock.json** is a lockfile containing exact dependencies being used

```
{  
  "name": "demo",  
  "version": "0.1.0",  
  "lockfileVersion": 2,  
  "requires": true,  
  "packages": {  
    "": {  
      "name": "demo",  
      "version": "0.1.0",  
      "dependencies": {  
        "react": "^18.2.0",  
        "react-dom": "^18.2.0",  
        "react-scripts": "5.0.1",  
      },  
      "devDependencies": {  
        "chai": "^4.3.7",  
        "chai-dom": "^1.11.0",  
        "mocha": "^10.2.0"  
      }  
    },  
    "node_modules/@adobe/css-tools": {  
      "version": "4.0.1",  
      "resolved": "https://registry.npmjs.org/@adobe/css-tools/-/css-tools-4.0.1.tgz",  
      "integrity": "sha512-+u76oB43n0HrF4DDWRLWDCtci7f3QJoEBigemIdIeTi10Dqjx6Tad9NCVnPRwewWlKkVab5PlK8D CtPTyX7S8g=="  
    },  
  }  
}
```



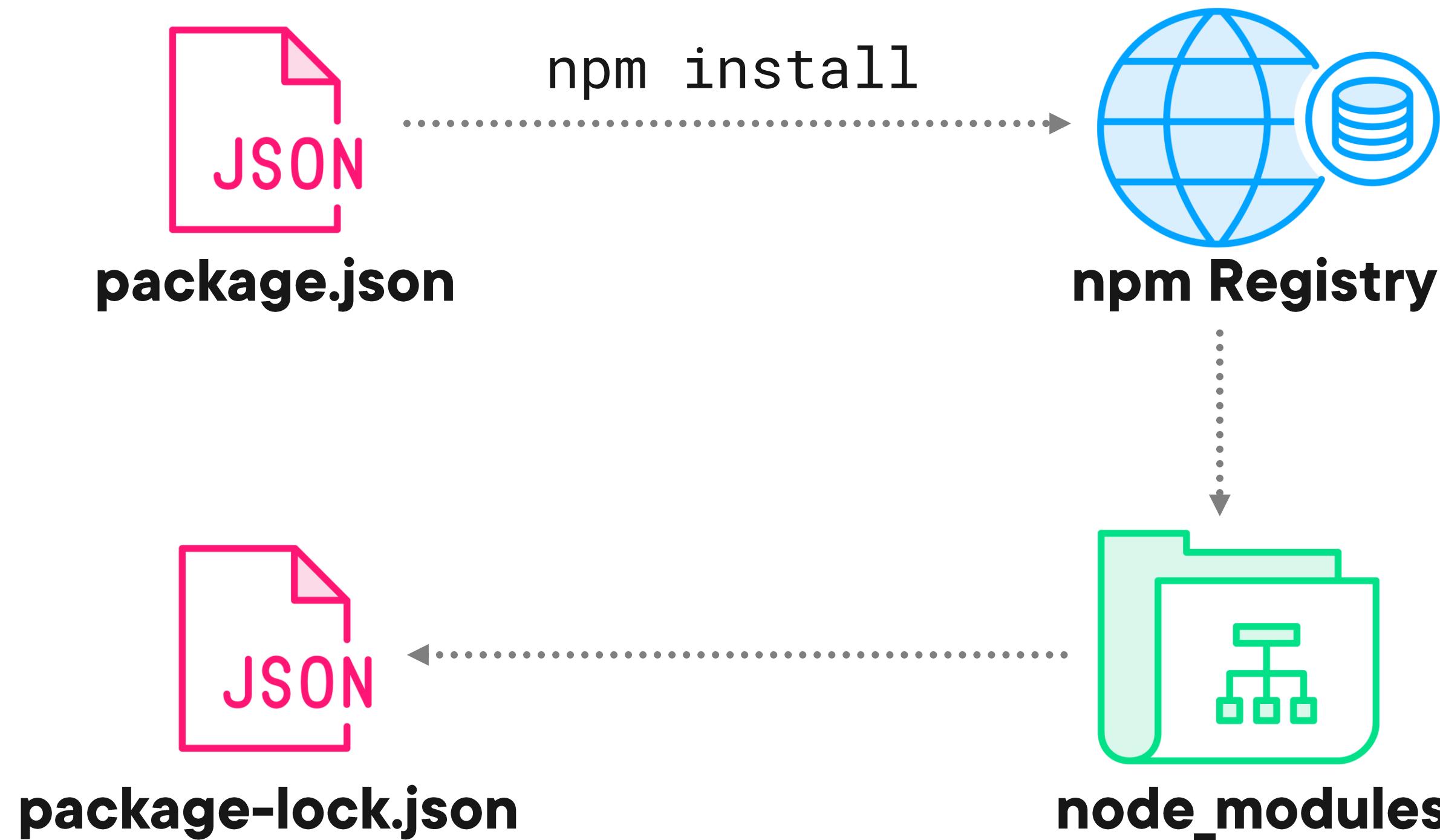
# The Veins of a JavaScript Project

The **node\_modules** folder contains the physical package files being used

Name	Date modified	Type	Size
.bin	5/25/2023 12:46 PM	File folder	
.cache	5/25/2023 1:05 PM	File folder	
@adobe	5/25/2023 12:45 PM	File folder	
@ampproject	5/25/2023 12:45 PM	File folder	
@babel	5/25/2023 12:45 PM	File folder	
@bcoe	5/25/2023 12:45 PM	File folder	
@cloudflare	5/25/2023 12:45 PM	File folder	
@csstools	5/25/2023 12:45 PM	File folder	
@emotion	5/25/2023 12:45 PM	File folder	
@eslint	5/25/2023 12:45 PM	File folder	
@humanwhocodes	5/25/2023 12:45 PM	File folder	
@istanbuljs	5/25/2023 12:45 PM	File folder	
@jest	5/25/2023 12:45 PM	File folder	
@jridgewell	5/25/2023 12:45 PM	File folder	
@leichtgewicht	5/25/2023 12:45 PM	File folder	
@nicolo-ribaudo	5/25/2023 12:45 PM	File folder	



# How npm Works



# Initializing a JavaScript Project



# Demo

**Initializing a project using npm install**  
**Creating a project using npm init**



```
npm install
```

## Installing Project Dependencies

**Execute the npm install command to pull down and extract the dependencies for a project**



**There are other JavaScript  
package managers besides  
npm.**



```
npm init
```

## Creating a New package.json File

**Use npm init to create a new JavaScript package definition in the current directory**



# npm init

```
package name: (<folder name>)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
```



# Demo: package.json



# Adding a Package Dependency



# Demo

**Install the latest version**

**Install a specific version**

**Install as a development dependency**



# Demo: VS Code Project



# Demo: yargs

yargs [DT](#)  
17.7.2 • Public • Published a year ago

[Readme](#) [Code](#) Beta [7 Dependencies](#) [0 Dependents](#) [250 Versions](#)



## Yargs

Yargs be a node.js library fer hearties tryin' ter parse optstrings

[CI](#) passing [npm](#) v17.7.2 [code style](#) standard [min coverage](#) 96% [Conventional Commits](#) 1.0.0 [Slack](#)

Install  
`> npm i yargs`

Repository  
[github.com/yargs/yargs](#)

Homepage  
[yargs.js.org/](#)

Weekly Downloads  
80,632,244 

Version  
17.7.2

License  
MIT



```
npm install yargs
```

```
npm install yargs --save
```

## Install the Latest Version

**Use npm install to install a runtime dependency. Adding the `--save` flag will forcefully add it to the package.json dependencies list.**



```
npm install <package>@<version>
```

```
# Example: Latest matching version  
npm install express@5
```

```
# Example: Specific version  
npm install express@5.1.0
```

## Install a Specific Version

**Use the `@` separator to specify a version of a package you want to install. The version string will be saved to the package.json.**



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

```
npm install <package> -D
```

## Install a Development Dependency

**Use the `--save-dev` flag to specify the dependency is used only for development-time. It will be saved to the `devDependencies` in the package.json.**



# Why Separate Development Dependencies?

They won't be installed in a production Node.js environment

They help teammates know what packages are not used at runtime



# Resources

**Specifying dependencies and devDependencies**

<https://docs.npmjs.com/specifying-dependencies-and-devdependencies-in-a-package-json-file>



# Updating a Package Dependency



# Demo

**List outdated dependencies**

**Perform a package upgrade**

**Discuss version constraints**

**When to use automation**



# Demo: VS Code Project



```
npm outdated
```

## **Listing Outdated Packages**

**Use the `outdated` command to list package current, wanted, and latest versions.**



`^17.7.0` – Any version with a major version of 17, minor of 7, and patch > 0

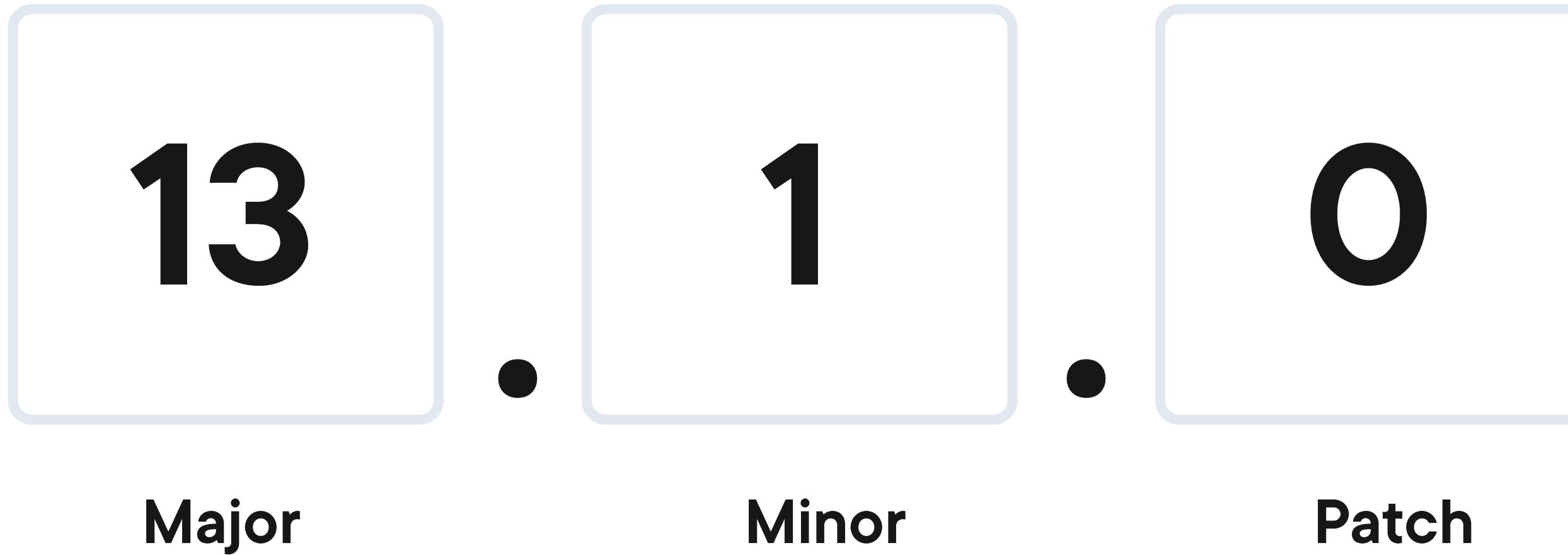
`17` – Any version with a major version of 17

## Matching Versions

**Use the update command to automatically perform a bulk update**



# Semantic Versioning



```
npm update --dry-run
```

## Doing an Update Dry Run

**Use the update command with a dry run flag to see what npm will do**



**npm update will only update  
to the highest-supported  
wanted version, not the  
latest version.**



```
npm update
```

## Doing an Update

**Use the update command to bulk update packages to the highest wanted version**



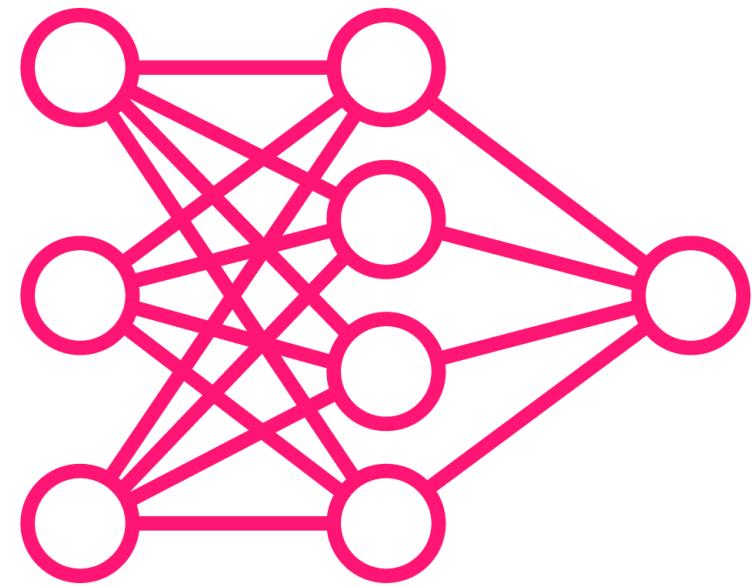
```
npm install@latest
```

## Update to the Latest Version

Use the "latest" version tag which always points to the latest published version.



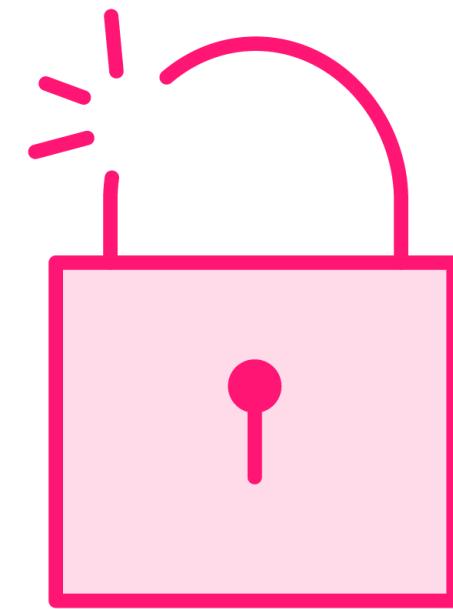
# Challenges of Updating Packages



**Complex to manage  
many dependencies**



**Newer versions aren't  
always backwards-  
compatible**



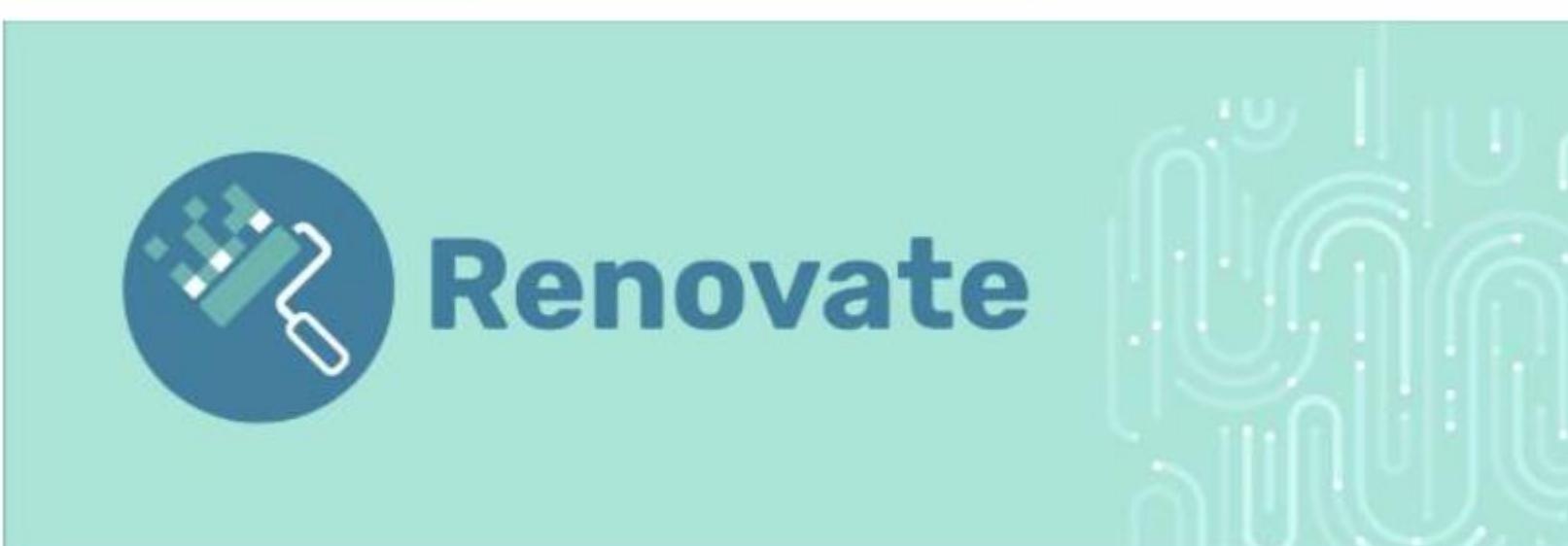
**Security  
vulnerabilities can be  
found downstream**



# B-Roll: Renovate Bot

**renovate**  
37.351.2 • Public • Published 4 hours ago

[Readme](#) [Code](#) [Beta](#) [105 Dependencies](#) [11 Dependents](#) [8,365 Versions](#)



**Renovate**

Automated dependency updates. Multi-platform and multi-language.

license AGPL-3.0-only codecov 100% renovate enabled Build passing docker pulls 1.4G  
openssf scorecard 8.2

Install  
`> npm i renovate`

Repository  
[github.com/renovatebot/renovate](https://github.com/renovatebot/renovate)

Homepage  
[renovatebot.com](https://renovatebot.com)

Weekly Downloads  
119,443

Version  
37.351.2

License  
AGPL-3.0-only



# Resources

**About semantic versioning**

<https://docs.npmjs.com/about-semantic-versioning>

**npm SemVer Calculator**

<https://semver.npmjs.com/>



# Removing a Package Dependency



# Demo

**Using npm to uninstall a package**

**Manually removing a package**



# Demo: VS Code Project



```
npm uninstall chalk
```

## Uninstall a Package

**Completely removes a package, updating the package.json and package-lock.json files accordingly.**



# Demo: Manually Removing a Package





# How IDEs Integrate with npm



# How IDEs Integrate with npm



# How IDEs Integrate with npm

Package dependency  
management



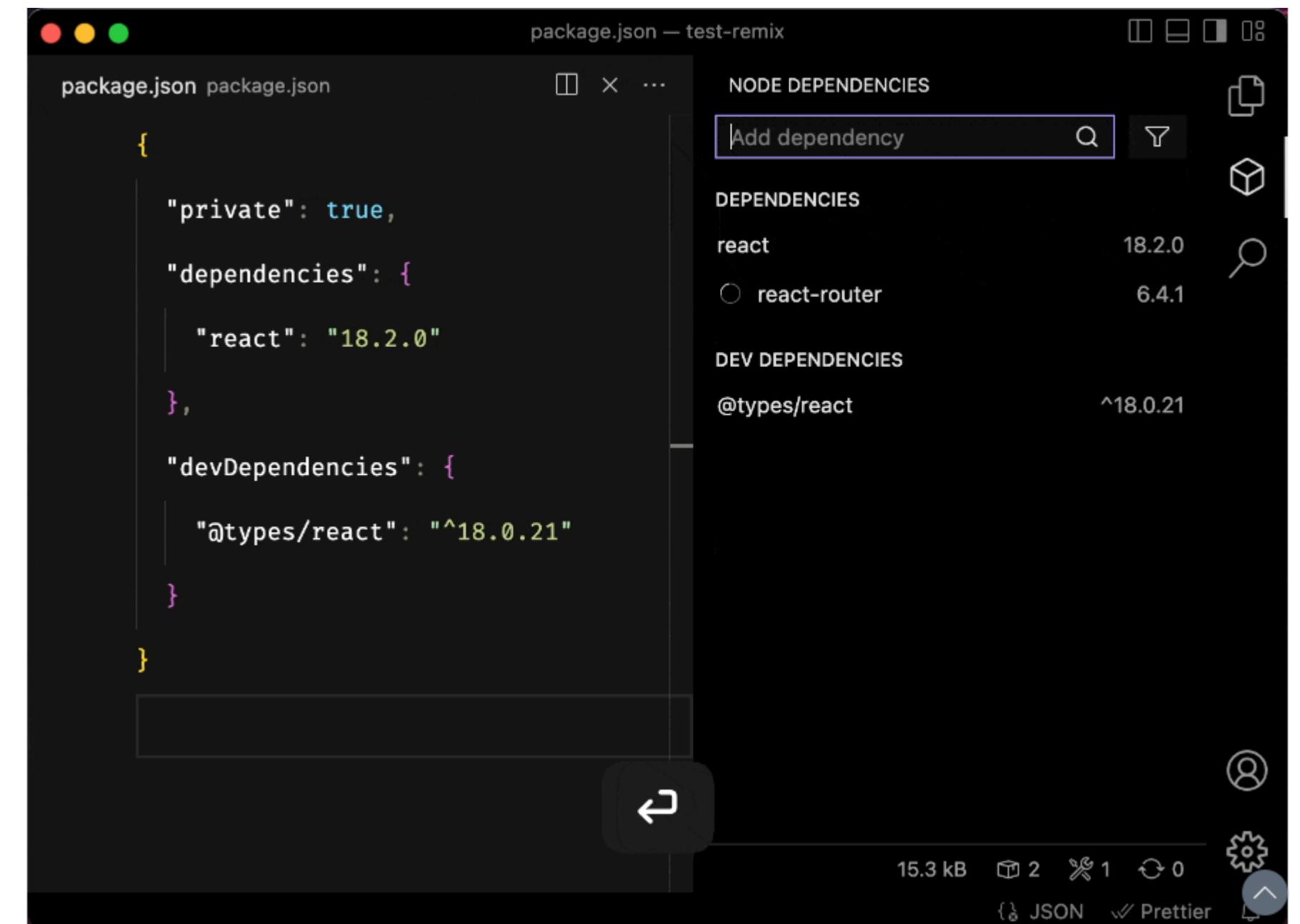
# Demo: Package Dependency Metadata

```
① package.json > ...
1  {
2    "name": "pluralsight-course-building-js-dev-environment
-demo",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
  ▶ Debug
6    "scripts": {
7      "test": "echo \\\"Error: no test
specified\\\" && exit 1"
8    }, Fast, unopinionated, minimalist web framework
9    "a
10   "l Latest version: 4.19.2 published 1 month ago
11   "d http://expressjs.com/
12     "express": "^4.19.2"
13   }
14 }
15 }
```



# Demo: npm VS Code Extension

```
package.json > ...
1  {
2    "name": "pluralsight-course-building-js-dev-environment"
3    "-demo",
4    "version": "1.0.0",
5    "description": "",
6    "main": "index.js",
7    > Debug
8    "scripts": {
9      "test": "echo \\\"Error: no test
10      specified\\\" && exit 1"
11    },
12    "dependencies": {
13      "express": "4.19.2"
14    }
15 }
```



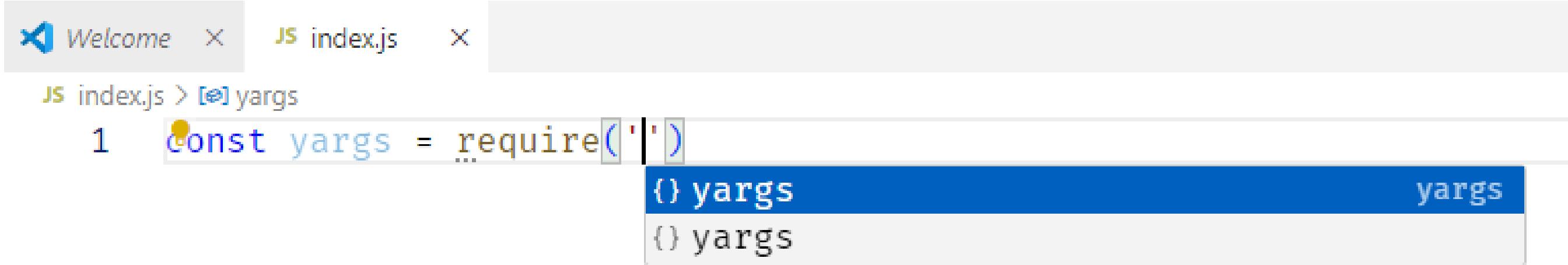
# How IDEs Integrate with npm

**Package dependency  
management**

**Provide code  
completion for  
modules**



# Demo: CommonJS Module Support



A screenshot of a code editor window titled "Welcome" with two tabs: "index.js". The code editor shows the following code:

```
1 const yargs = require('')
```

The cursor is positioned at the end of the line after the opening parenthesis of the require statement. A tooltip or dropdown menu is open, displaying the result of the module resolution:

- {} yargs
- {} yargs

The second item in the list, "yargs", is highlighted with a blue background, indicating it is the selected module definition.



# How IDEs Integrate with npm

**Package dependency management**

**Provide code completion for modules**

**Informs language tooling versions and configuration**



# Demo: jsconfig.json

Configures editor tooling and Intellisense for VS Code

```
{  
  "compilerOptions": {  
    "module": "CommonJS",  
    "moduleResolution": "Node",  
    "target": "ES2015",  
  },  
  "typeAcquisition": {  
    "include": ["node"]  
  },  
  "exclude": [  
    "node_modules",  
    "**/node_modules/*"  
  ]  
}
```



# Demo: CommonJS Syntax



# JavaScript IDE Configuration

## Visual Studio Code

<https://code.visualstudio.com/docs/nodejs/working-with-javascript>

## JetBrains WebStorm

<https://www.jetbrains.com/help/webstorm/javascript-specific-guidelines.html>

## Eclipse JSDT Environment Setup

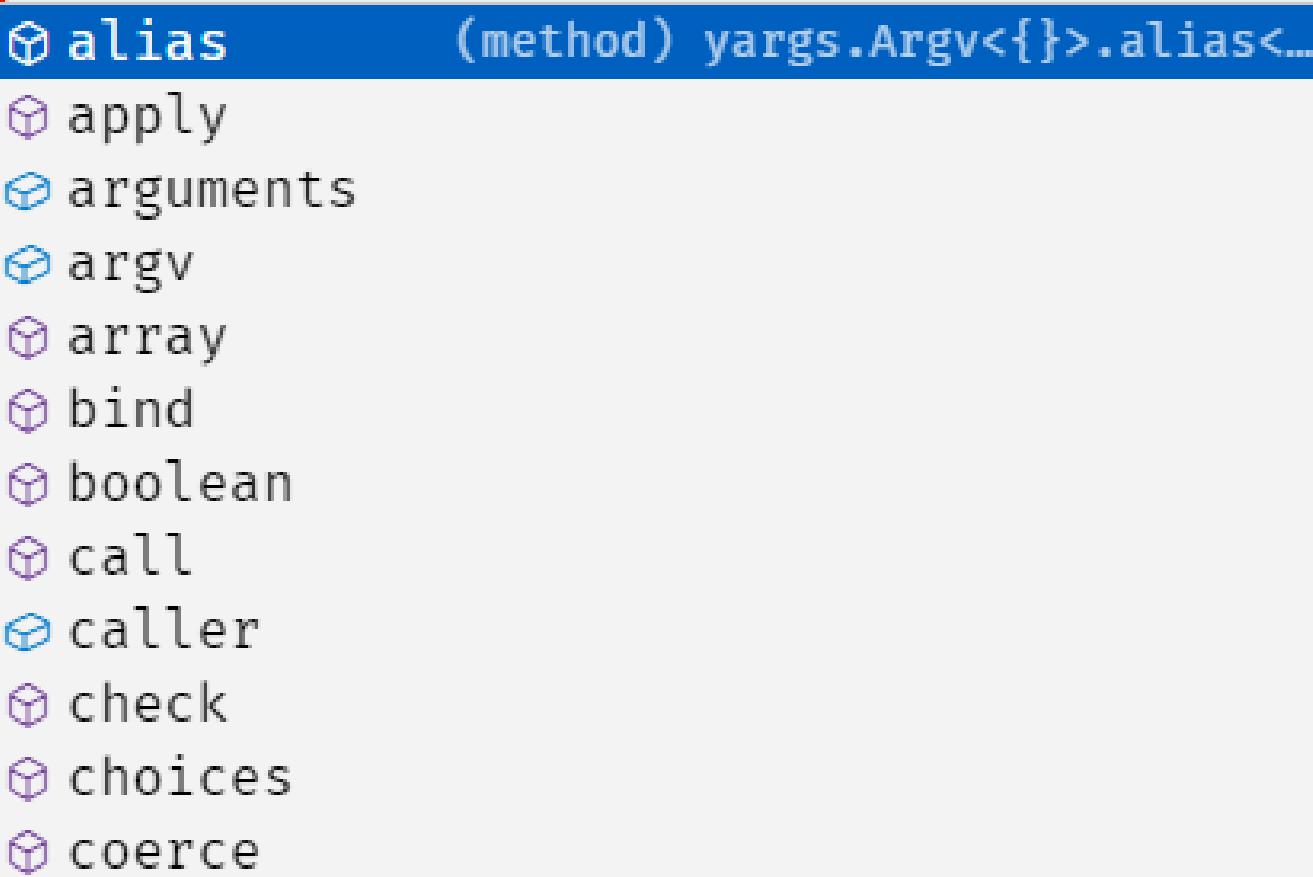
<https://www.youtube.com/watch?v=SFxfuudlau8>



# Demo: Type Acquisition

JS index.js > ...

```
1 const yargs = require('yargs');
2
3 yargs.
```



# How IDEs Integrate with npm

**Package dependency management**

**Provide code completion for modules**

**Informs language tooling versions and configuration**

**Automatic type acquisition**



# Demo: Node Typings

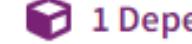
Additional typings can be acquired using `jsconfig.json`

```
{  
  "compilerOptions": {  
    "module": "CommonJS",  
    "moduleResolution": "Node",  
    "target": "ES2015",  
  },  
  "typeAcquisition": {  
    "include": ["node"]  
  },  
  "exclude": [  
    "node_modules",  
    "**/node_modules/*"  
  ]  
}
```



# Demo: @types/ Packages

**@types/node **  
20.12.11 • Public • Published a day ago

 Readme  Code  Beta  1 Dependency  36,452 Dependents  1,824 Versions

---

## Installation

```
npm install --save @types/node
```

---

## Summary

This package contains type definitions for node (<https://nodejs.org/>).

## Details

Files were exported from  
<https://github.com/DefinitelyTyped/DefinitelyTyped/tree/master/types/node>.

Additional Details

Install

```
> npm i @types/node
```

---

Repository

↳ [github.com/DefinitelyTyped/DefinitelyT...](https://github.com/DefinitelyTyped/DefinitelyTyped/tree/master/types/node)

---

Homepage

🔗 [github.com/DefinitelyTyped/DefinitelyT...](https://github.com/DefinitelyTyped/DefinitelyTyped)

---

Weekly Downloads

99,094,092



---

Version

20.12.11

License

MIT



# Demo: Node Type Annotations

```
JS index.js > ...
1 const yargs = require('yargs');
2
3 yargs.]
  ↴ alias      (method) yargs.Argv<{}>.alias<...
  ↴ apply
  ↴ arguments
  ↴ argv
  ↴ array
  ↴ bind
  ↴ boolean
  ↴ call
  ↴ caller
  ↴ check
  ↴ choices
  ↴ coerce
```



# How IDEs Integrate with npm

**Package dependency management**

**Provide code completion for modules**

**Informs language tooling versions and configuration**

**Automatic type acquisition**

**Detect problems and errors**



# Demo: Error Annotations

```
JS index.js > [o]options
1 const yargs = require("yargs");
2
3 const options
File is a CommonJS module; it may be converted to an ES module. ts(80001)
var require: NodeRequire
(id: string) => any
Quick Fix... (Ctrl+.)
```



# Demo: Problems Panel



You aren't required to fix  
IDE errors, but they may  
anticipate runtime errors.



# Demo: Run with Invalid Syntax



# How IDEs Integrate with npm

**Package dependency management**

**Provide code completion for modules**

**Informs language tooling versions and configuration**

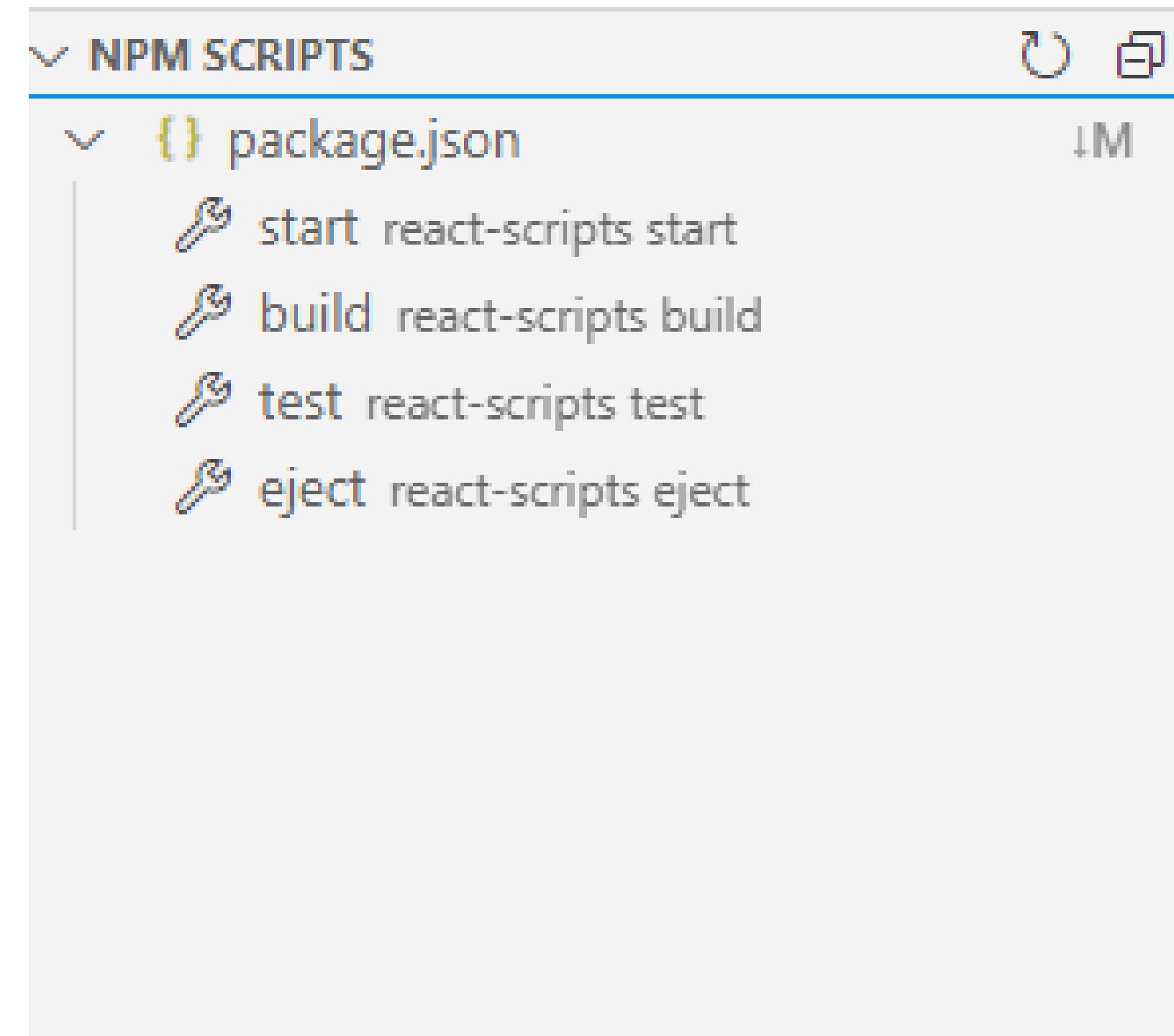
**Automatic type acquisition**

**Detect problems and errors**

**Run npm scripts and tasks**



# Demo: Run npm Scripts



# How IDEs Integrate with npm

**Package dependency management**

**Provide code completion for modules**

**Informs language tooling versions and configuration**

**Automatic type acquisition**

**Detect problems and errors**

**Run npm scripts and tasks**



# Troubleshooting npm Issues



# Troubleshooting

**Fixing peer dependency errors**

**Dealing with corporate proxies**

**Random package installation failures**



**ERESOLVE** unable to resolve dependency tree

## Peer Dependency Error

**Some packages require other packages to be at a certain version range, otherwise npm will throw a peer dependency error**



```
npm install <package> --force
```

```
npm install <package> --legacy-peer-deps
```

## Fixing Peer Dependency Errors

**Ultimately the package maintainers need to update their version range but there are ways to workaround it if you know the packages are compatible**



# Demo: Check npm Package Website



# **Cannot Connect to npm Registry**

**Create a .npmrc file and configure your registry settings. Configure any proxy or firewall settings.**



```
> .npmrc
```

```
registry=https://npmjs.com
```

## Cannot Connect to npm Registry

Create a `.npmrc` file and configure your registry settings. Configure any proxy or firewall settings.



# Demo: Configuring npm

[CLI](#) / [Using](#) / Registry

## registry

The JavaScript Package Registry

Select CLI Version:

Version 8.19.4 (Legacy) ▾

### Description

To resolve packages by name and version, npm talks to a registry website that implements the [Common Package Registry](#) specification for reading package info.

npm is configured to use the **npm public registry** at <https://registry.npmjs.org> by default. Use [this link](#) to switch to another registry. The public registry is subject to terms of use available at <https://docs.npmjs.com/policies/terms>.

[CLI](#) / [Configuring](#) / [.npmrc](#)

## .npmrc

The npm config files

Select CLI Version:

Version 8.19.4 (Legacy) ▾

### Description

npm gets its config settings from the command line, environment variables, and `.npmrc` files. The `npm config` command can be used to update and edit the contents of the user and global `npmrc` files. For a list of available configuration options, see [config](#).



```
rm -rf node_modules
```

## Reset Node Modules

**Sometimes your `node_modules` folder will get into an inconsistent state. The nuclear option is to completely remove the directory.**



```
code EPROTO  
errno EPROTO  
request to https://registry.npmjs.org/package failed,  
reason: write EPROTO...
```

## Error When Pulling down a npm Package

HTTP errors can sometimes happen when packages change in the registry over time



```
npm cache clean --force
```

```
npm cache verify
```

## Clean npm Cache

For even harder issues, you can try cleaning the npm cache on your system (in addition to removing node\_modules).

<https://stackoverflow.com/a/48826188>



# Troubleshooting Resources

**Troubleshooting Common npm Errors**

<https://docs.npmjs.com/common-errors>

**StackOverflow Questions Tagged "npm"**

<https://stackoverflow.com/questions/tagged/npm>



# Summary

**npm manages dependencies for both browser- and Node.js-based JavaScript projects**

**The command you'll use the most day-to-day is npm install, in all its variations**

**Understanding package versioning constraints will demystify most issues**

**Your IDE is a powerful tool for making you more productive with coding assistance**

**You'll run into problems, so it pays to know how to deal with them**

