What is the difference between --save and --save-dev?

Asked 5 years, 7 months ago Active 1 month ago Viewed 177k times

617

134

What is the difference between:

> npm install [package_name] --save

and

> npm install [package_name] --save-dev

What does this mean?

node.js npm save package

edited Mar 31 at 4:08

ivanleoncz
2,833 3 30 39

asked Apr 6 '14 at 7:34

nfort
3,208 3 9 12

3 yeah I am confused about this - if you use continuous integration like Jenkins, does Jenkins know to use the devDependencies modules for running tests? I assume so but it's not super obvious. – Alexander Mills Aug 12 '15 at 19:32

4 perhaps edit the question to also say, what is the functional difference between dependencies and devDependencies? – Alexander Mills Aug 12 '15 at 19:33

3 Packages installed via the --save-dev option are not re-installed when the user executes `npm install --production` . That's the operational difference (see https://docs.npmjs.com/cli/install for more info). – Andrew Jul 20 '17 at 14:03

7 @MuhammadUmer That's precisely why people ask questions on here - in order to 'get a clue'. Perhaps adding a real answer would be more productive - this is definitely an interesting distinction that I was not aware of. – Simon_Weaver Jan 1 '18 at 20:37

2 also if you set environment variable `NODE_ENV` to production, then just `npm install` automatically excludes development packages. – Muhammad Umer Jan 1 '18 at 22:58

12 Answers

493

- `--save-dev` is used to save the package for development purpose. Example: unit tests, minification..
- `--save` is used to save the package required for the application to run.

answered Jul 11 '15 at 15:56

Tuong Le
11k 5 41 42

130 How are they different? When would I use one vs the other? Can I still use it the package in production if it is under --save-dev? – Dave Voyles - MSFT Nov 8 '16 at 15:30

13 The answer succinctly answers your first two questions. The answer to the last question, "Can I still use the package in production if it is under --save-dev," is "no." While it's certainly *possible* to do this, it is not intended. – Technetium Jan 18 '17 at 18:33

54 Shorthand versions: `-D` is short for `--save-dev` and `-S` is short for `--save` – chrisco Mar 26 '17 at 14:07

122 This answer is frustratingly vague. Even a small example would go a long way to helping make this clearer. – Choylton B. Higginbottom Oct 31 '17 at 17:45

24 Note that as of npm version 5.0.0, the `--save` option is no longer necessary. If you do `npm install my-package` , it will add "my-package" as a dependency in the package.json file. – Martin Carel Feb 23 '18 at 23:27

543

The difference between `--save` and `--save-dev` may not be immediately noticeable if you have tried them both on your own projects. So here are a few examples...

Lets say you were building an app that used the **moment** package to parse and display dates. Your app is a scheduler so it really needs this package to run, as in: **cannot run without it**. In this case you would use

```
npm install moment --save
```

This would create a new value in your package.json

```
"dependencies": {
   ...
   "moment": "^2.17.1"
}
```

When you are developing, it really helps to use tools such as test suites and may need jasmine-core and karma. In this case you would use

```
npm install jasmine-core --save-dev
npm install karma --save-dev
```

This would also create a new value in your package.json

```
"devDependencies": {
    ...
    "jasmine-core": "^2.5.2",
    "karma": "^1.4.1",
}
```

You do **not need** the test suite to run the app in its normal state, so it is a `--save-dev` type dependency, nothing more. You can see how if you do not understand what is really happening, it is a bit hard to imagine.

Taken directly from NPM docs docs#dependencies

> **Dependencies**
>
> Dependencies are specified in a simple object that maps a package name to a version range. The version range is a string which has one or more space-separated descriptors. Dependencies can also be identified with a tarball or git URL.
>
> **Please do not put test harnesses or transpilers in your dependencies object. See devDependencies**, below.

Even in the docs, it asks you to use --save-dev for modules such as test harnesses.

I hope this helps and is clear.

edited Aug 10 at 11:51

Aleksandar Belic
129 3 12

answered Feb 13 '17 at 14:25

Michael Bruce
5,853 1 16 22

10 IMO, i think the 'save' keyword is a problem. Why don't they make -dev flag for develop and -deploy for deployment. It make sense than 'save' keyword. – Thinh Vu Mar 17 '17 at 7:44

Why doesn't the package just know (decide) if it's a release package or a dev package and --save be used for both. Seems odd to make the installing user decide this, when the package developer creates the intent. – CodeGrue Jun 4 '17 at 1:21

4 CodeGrue, if you use jQuery only for testing React components it would go in save-dev, but you may not actually use it to build your main project. Yes, this is possible. So why would the packager know what you are doing with it? – Michael Bruce Jun 5 '17 at 23:04

2 Much clearer. I'm an embedded guy learning Bootstra + Node.js workflow for the first time. It's not obvious what the difference is off the cuff. – Leroy105 Mar 2 '18 at 0:24

2 @YakovL save-dev means the packages are not installed when somebody else installs your package as their dependency. Packages that are only used to run scripts such as start/build will not be needed in that case, so they're put in dev-dependencies. If you're working on a web app and not on a package for use by others, you probably shouldn't worry about it at all. – riv Jun 6 at 8:27

86

By default, NPM simply installs a package under node_modules. When you're trying to install dependencies for your app/module, you would need to first install them, and then add them to the `dependencies` section of your `package.json`.

`--save-dev` adds the third-party package to the package's development dependencies. It won't be installed when someone installs your package. It's typically only installed if someone **clones** your source repository and runs `npm install` in it.

`--save` adds the third-party package to the package's dependencies. It will be installed together with the package whenever someone runs `npm install package`.

Dev dependencies are those dependencies that are only needed for developing the package. That can include test runners, compilers, packagers, etc. Both types of dependencies are stored in the package's `package.json` file. `--save` adds to `dependencies`, `--save-dev` adds to `devDependencies`

npm install documentation can be referred here.

answered Feb 15 '17 at 11:17

Lakshmi Swetha G
1,213 5 13

27 I suspected this... you can use --save-dev and --save interchangeably if you are build a web app that won't become a package i.e. downloaded from npm, if you are developing a package to share with others, it is important to understand the difference. – VFein Mar 31 '17 at 13:58

8 Thank you finally someone that says its purpose when you use npm install – CapturedTree Sep 10 '17 at 2:37

--save is now default with npm install with the release of npm 5 in 2017 – Natalie Jul 11 at 19:56

wait, why complex sentences? In DevDependecy developer can install the packages, and it will be updated the devDevependency only. So when new developer clone the project codebase and run npm install => here only `dependency package name is going to install.` in node_modules.. not developer's package as in Dev-dependency. – Anupam Maurya Sep 24 at 11:51

A perfect example of this is:

53

```
$ npm install typescript --save-dev
```

In this case, you'd want to have Typescript (a javascript-parseable coding language) available for development, but once the app is deployed, it is no longer necessary, as all of the code has been transpiled to javascript. As such, it would make no sense to include it in the published app. Indeed, it would only take up space and increase download times.

edited Jun 30 '17 at 16:13

answered Jun 18 '17 at 19:11

Jackalope

1,349 16 29

3 The same goes for: "$ npm install grunt --save-dev", as it is useful for development, but not for deployment. – Jackalope Jun 18 '17 at 19:15

1 A side note: Microsoft suggests installing @types/xxx packages as dependencies, not devDependencies github.com/Microsoft/types-publisher/issues/81 – Dave Nov 6 '17 at 11:44

2 What I find confusing is how does this even matter? Packages saved using `--save` are still only saved in the `node_modules` folder. The code isn't included in the deployed website. – Kokodoko Nov 8 '17 at 15:12

5 @Kokodoko When you use the `--save-dev` flag, the package is added to your `devDependencies` object. If/when someone installs **your** package, all the `dependencies` are downloaded but the `devDependencies` are not, since they aren't required at runtime. As the answer stated, this saves them time and space. Developers working on your package files itself can just run `npm install` inside the package directory to install the `devDependencies` as well. – Jasjit Singh Marwah Mar 18 '18 at 7:18

So if you download a repo from github and type `npm install` , the `devDependencies` are ignored? – Kokodoko Mar 19 '18 at 9:00

As suggested by @andreas-hultgren in this answer and according to the npm docs:

33

> If someone is planning on downloading and using your module in their program, then they probably don't want or need to download and build the external test or documentation framework that you use.

However, for webapp development, Yeoman (a scaffolding tool that installs a peer-reviewed, pre-written package.json file amongst other things) places all packages in devDependencies and nothing in dependencies, so it appears that the use of `--save-dev` is a safe bet in *webapp* development, at least.

edited May 23 '17 at 12:10

answered Apr 14 '14 at 8:32


Community ♦
1 1


wayfarer_boy
1,209 18 21

3 Note that I've run into issues when using gulp and installing packages with `--save-dev` where the package would not install its required dependancies. Running `--save` installed those missing dependancies. – Nick M Mar 3 '15 at 3:52

17 I'd also like to note that I'm now using `--save` for all but test and documentation dependencies (as per the npm docs). I'm beginning the think the Yeoman example I mentioned above is *not* a good example of best practice. – wayfarer_boy Mar 3 '15 at 9:17

I think so too, why would you ever need `--save-dev` is only becoming less clear with every answer here :) – Kokodoko Nov 8 '17 at 15:16

18

`--save-dev` saves semver spec into "devDependencies" array in your package descriptor file, `--save` saves it into "dependencies" instead.

answered Apr 6 '14 at 8:07


alex
9,427 1 24 41

79 and what's the functional difference? – ahnbizcad Apr 4 '15 at 16:22

5 this answer makes the most sense to me, devDependencies are then required for development but not production, so htmllint, sass compilation etc and Dependencies are for production requirements, such as Diaporama that will need to be present for things to run. – miller the gorilla Jul 19 '16 at 9:37

3 @ahnbizcad It's answered better here but the primary functional difference is that devDependencies are not transitively included. – Pace Sep 8 '16 at 15:54

Isn't the most intuitive way to describe it for someone who doesn't already know, this?: Dev `--save-dev` makes packages local to your project, whereas `--save` makes them local to your installation of node? – ahnbizcad Sep 8 '16 at 17:34

11

Let me give you an example,

- You are a developer of a very **SERIOUS npm library**. Which uses different testing libraries to test the package.
- An user Downloaded your library and want to use it in their code. Do they need to download your testing libraries as well? Maybe you use `jest` for testing and they use `mocha` . Do you want them to install `jest` as well? *Just* To run your library?

No. right? That's why they are in `devDependencies` .

When someone does, `npm i yourPackage` only the libraries required to **RUN** your library will be installed. Other libraries you used to bundle your code with or testing and mocking will not be installed because you put them in `devDependencies` . Pretty neat right?

So, **Why** do the developers need to expose the **devDependancies**?

Let's say your package is an open source package and 100s of people are sending pull requests to your package. Then how they will test the package? They will `git clone` your repo and when they would do an `npm i` the **dependencies** as well as **devDependencies**. Because they are not using your package. They are developing the package further, thus, in order to test your package they need to pass the existing test cases as well write new. So, they need to use your `devDependencies` which contain all the testing/building/mocking libraries that YOU used.

edited Oct 9 at 8:04

answered Jul 1 at 12:16

Aritra Chakraborty
5,731 1 11 23

3 Much better than the accepted answer as well as the answer with the maximum votes as this answer is more practical in nature. Thanks! – Uncaught Exception Oct 9 at 7:20

This should be the accepted answer – Harry Oct 24 at 14:44

7

Clear answers are already provided. But it's worth mentioning how `devDependencies` affects installing packages:

> By default, npm install will install all modules listed as dependencies in package.json . With the --production flag (or when the NODE_ENV environment variable is set to production), npm will not install modules listed in devDependencies .

See: https://docs.npmjs.com/cli/install

answered Aug 13 '18 at 3:25

Alireza
7,060 3 31 50

6

You generally don't want to bloat production package with things that you only intend to use for Development purposes.

Use `--save-dev` (or `-D`) option to separate packages such as Unit Test frameworks (jest, jasmine, mocha, chai, etc.)

Any other packages that your app needs for Production, should be installed using `--save` (or `-S`).

```
npm install --save lodash       //prod dependency
npm install -S moment           // "       "
npm install -S opentracing      // "       "

npm install -D jest                 //dev only dependency
npm install --save-dev typescript   //dev only dependency
```

If you open the `package.json` file then you will see these entries listed under two different sections:

```
"dependencies": {
  "lodash": "4.x",
  "moment": "2.x",
  "opentracing": "^0.14.1"
},

"devDependencies": {
    "jest": "22.x",
    "typescript": "^2.8.3"
},
```

edited Mar 31 at 4:39

ivanleoncz

2,833 3 30 39

answered Jul 22 '18 at 19:26

velhala

121 1 4

5

--save-dev is used for modules used in development of the application,not require while running it in production envionment **--save** is used to add it in package.json and it is required for running of the application.

Example: express,body-parser,lodash,helmet,mysql all these are used while running the application use --save to put in dependencies while mocha,istanbul,chai,sonarqube-scanner all are used during development ,so put those in dev-dependencies .

npm link or npm install will also install the dev-dependency modules along with dependency modules in your project folder

answered Nov 28 '17 at 8:30

Biswadev

749 5 18

0

I want to add some my ideas as

I think all differents will appear when someone use your codes instead of using by yourself

For example, you write a HTTP library called `node's request`

In your library,

you used lodash to handle string and object, without lodash, your codes cannot run

If someone use your HTTP library as a part of his codes. Your codes will be compiled with his.

your codes need lodash, So you need put in `dependencies` to **compile**

If you write a project like `monaco-editor` , which is a web editor,

you have bundle all your codes and your `product env library` using webpack, when build completed, only have a `monaco-min.js`

So someone don't case whether `--save` or `--save-dependencies` , only he need is `monaco-min.js`

Summary:

1. If someone want to compile your codes (use as library), put `lodash` which used by your codes into `dependencies`
2. If someone want add more feature to your codes, he need `unit test` and `compiler` , put these into `dev-dependencies`

answered Oct 14 '18 at 2:50

toffee

908 1 5 18

0

People use npm on production to do wicked cool stuff, Node.js is an example of this, so you don't want all your dev tools being run.

If you are using gulp (or similar) to create build files to put on your server then it doesn't really matter.

answered Mar 19 at 10:53

Tristanisginger

458 8 27