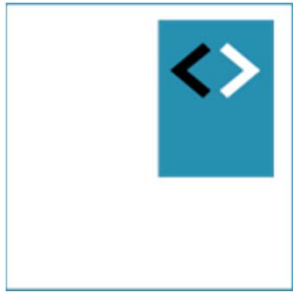


Node.js



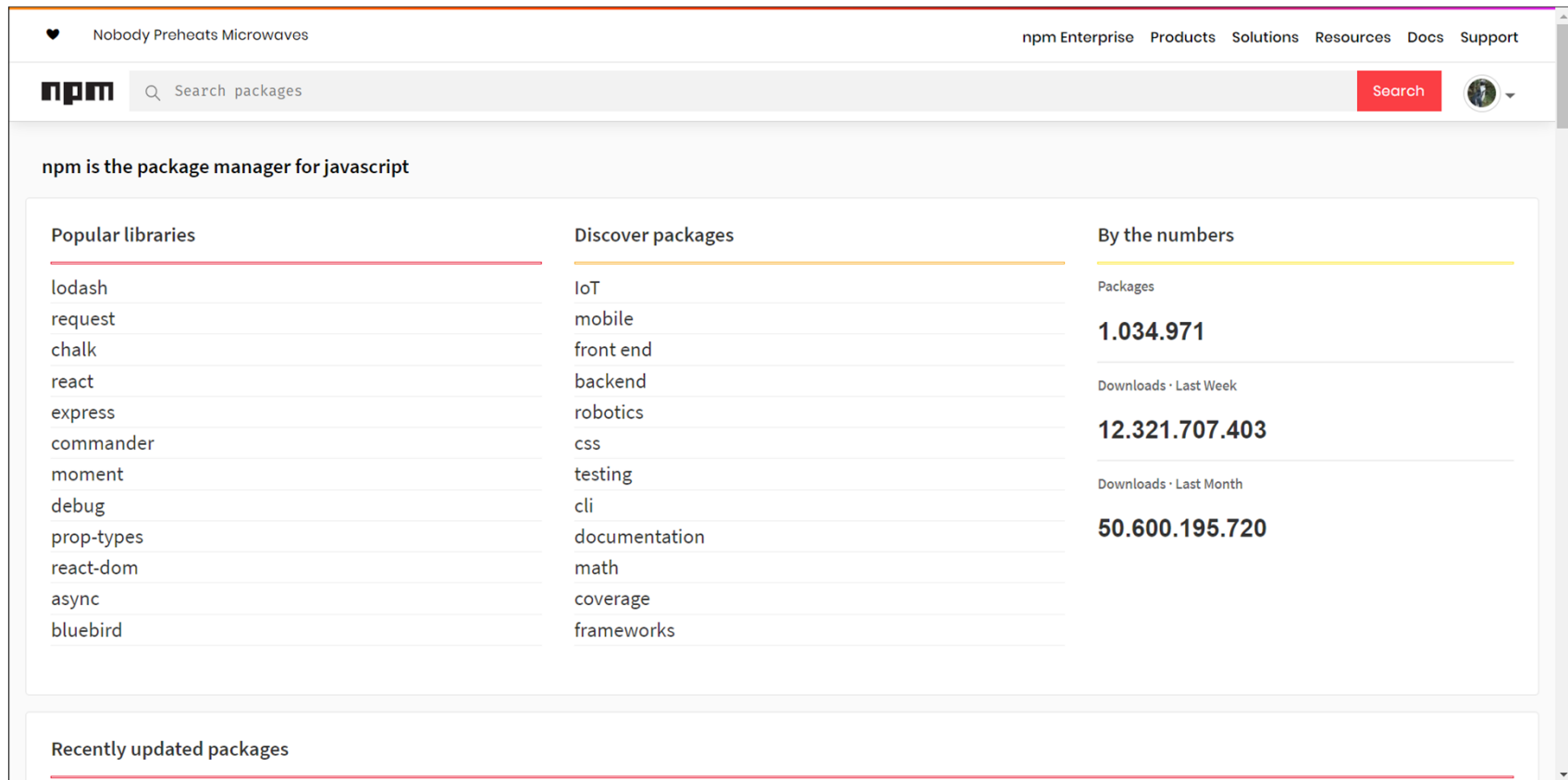
Peter Kassenaar

Module 5 – Node.js Deployment



Publishing to npm

Bringing your modules to the world

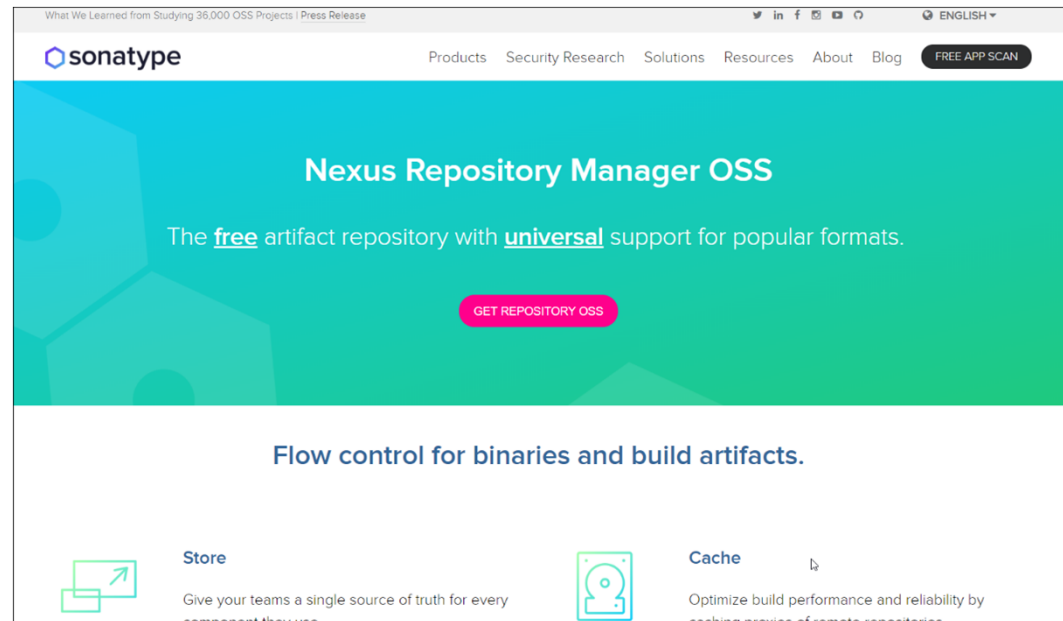


Npm consists of three parts:

- The website
- The Repository
- The Command Line Interface (CLI)

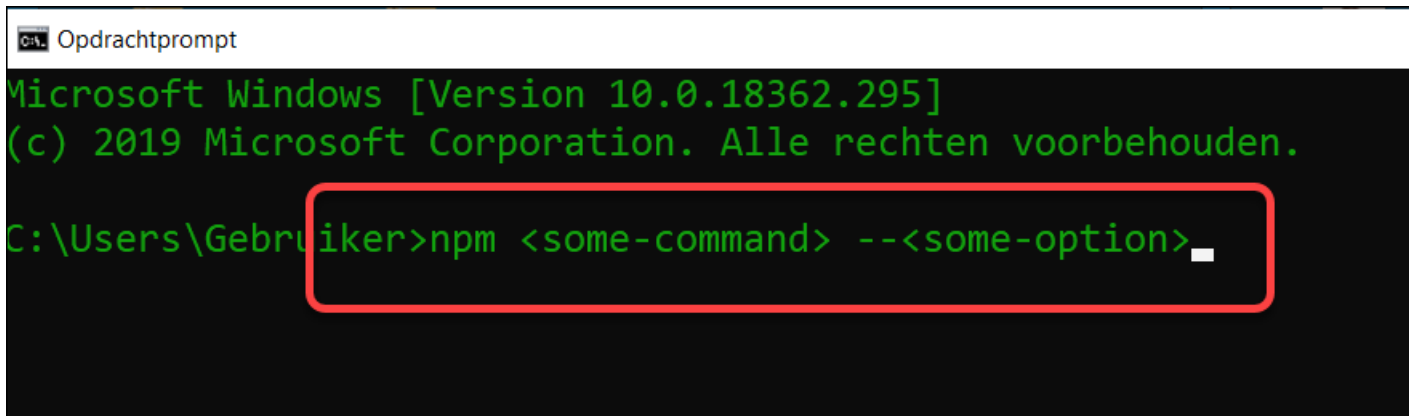
Working with npm

- In order to work with npm you need to know a little bit about all three.
- You can also work with **Nexus** – which can act as in-company, private version of the npm repository



The npm Command Line Interface

- Most developers will be working with the npm CLI
- It is installed when installing Node.js and available as a global command
- You can create an account with npm and upload ('publish') packages from your local computer

A screenshot of a Windows Command Prompt window titled "Opdrachtprompt". The window has a black background with green text. It displays the Microsoft Windows version (10.0.18362.295) and copyright information (© 2019 Microsoft Corporation). The current directory is "C:\Users\Gebruiker". The command prompt shows the syntax for running an npm command: "npm <some-command> --<some-option>". The command and its options are enclosed in a red rounded rectangle.

```
Opdrachtprompt
Microsoft Windows [Version 10.0.18362.295]
(c) 2019 Microsoft Corporation. Alle rechten voorbehouden.
C:\Users\Gebruiker>npm <some-command> --<some-option>_
```

Learn the CLI-commands and -flags

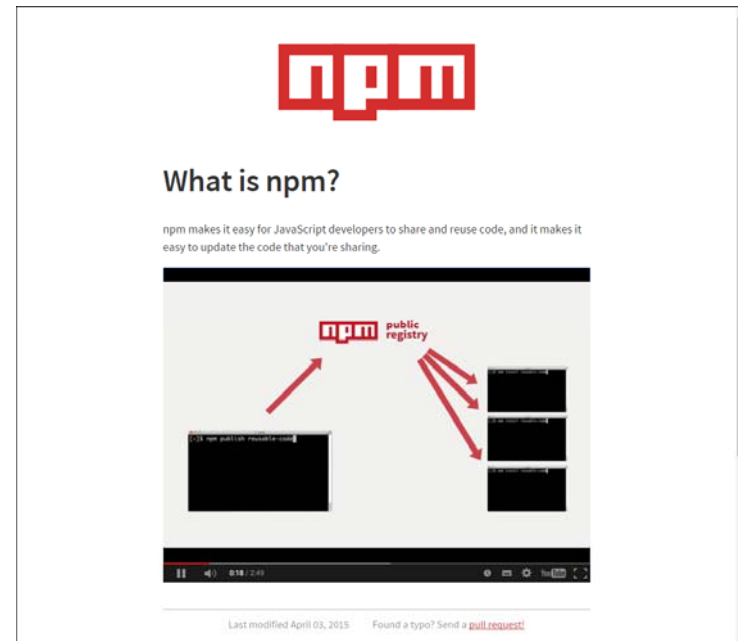
`npm install ...`

`npm uninstall ...`

`npm update ...`

`npm publish ...`

`npm adduser ...`



<https://docs.npmjs.com/cli-documentation/>

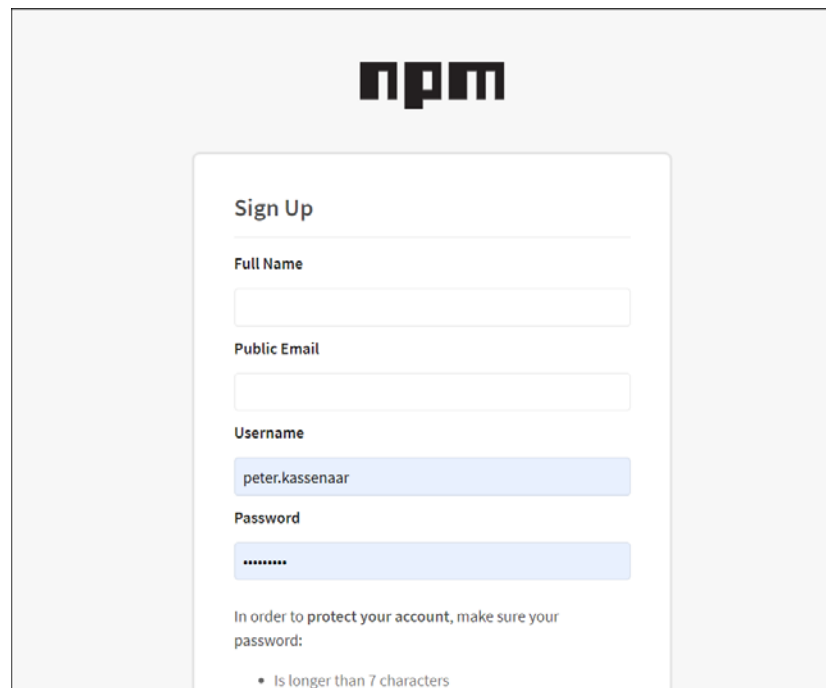
That's what we're going to do!

At least – the **most used**,
most important commands

Adding/creating your npm account

- First –create your account with npm:

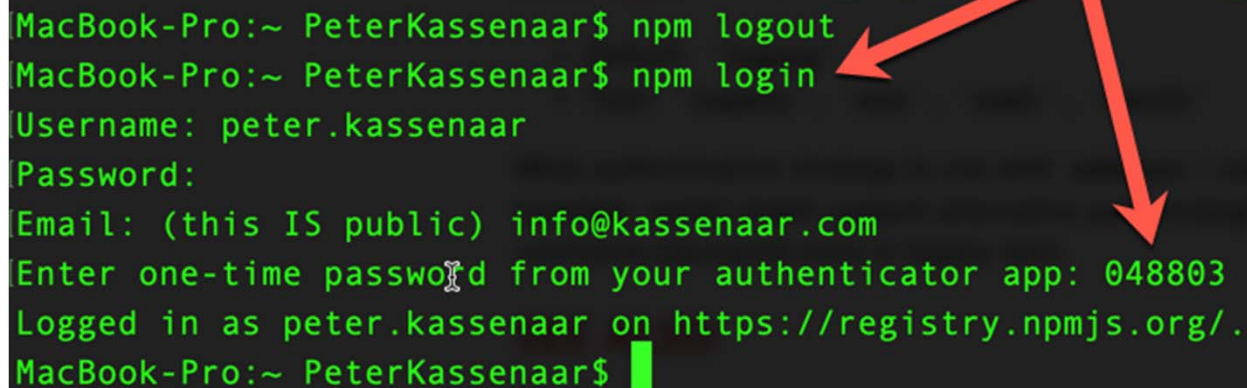
<https://www.npmjs.com/signup>



The image shows a screenshot of the npm Sign Up page. At the top, the npm logo is displayed. Below it, the title "Sign Up" is centered. The form contains four input fields: "Full Name", "Public Email", "Username", and "Password". The "Username" field is filled with "peter.kassenaar" and the "Password" field is filled with "*****". Below the password field, there is a note: "In order to protect your account, make sure your password:" followed by a bullet point: "• Is longer than 7 characters".

Log in

- Local machine: `npm adduser`
 - Add credentials as provided at sign up
- OR: `npm login`
 - Pass credentials
 - 2FA depends on what you set up yourself

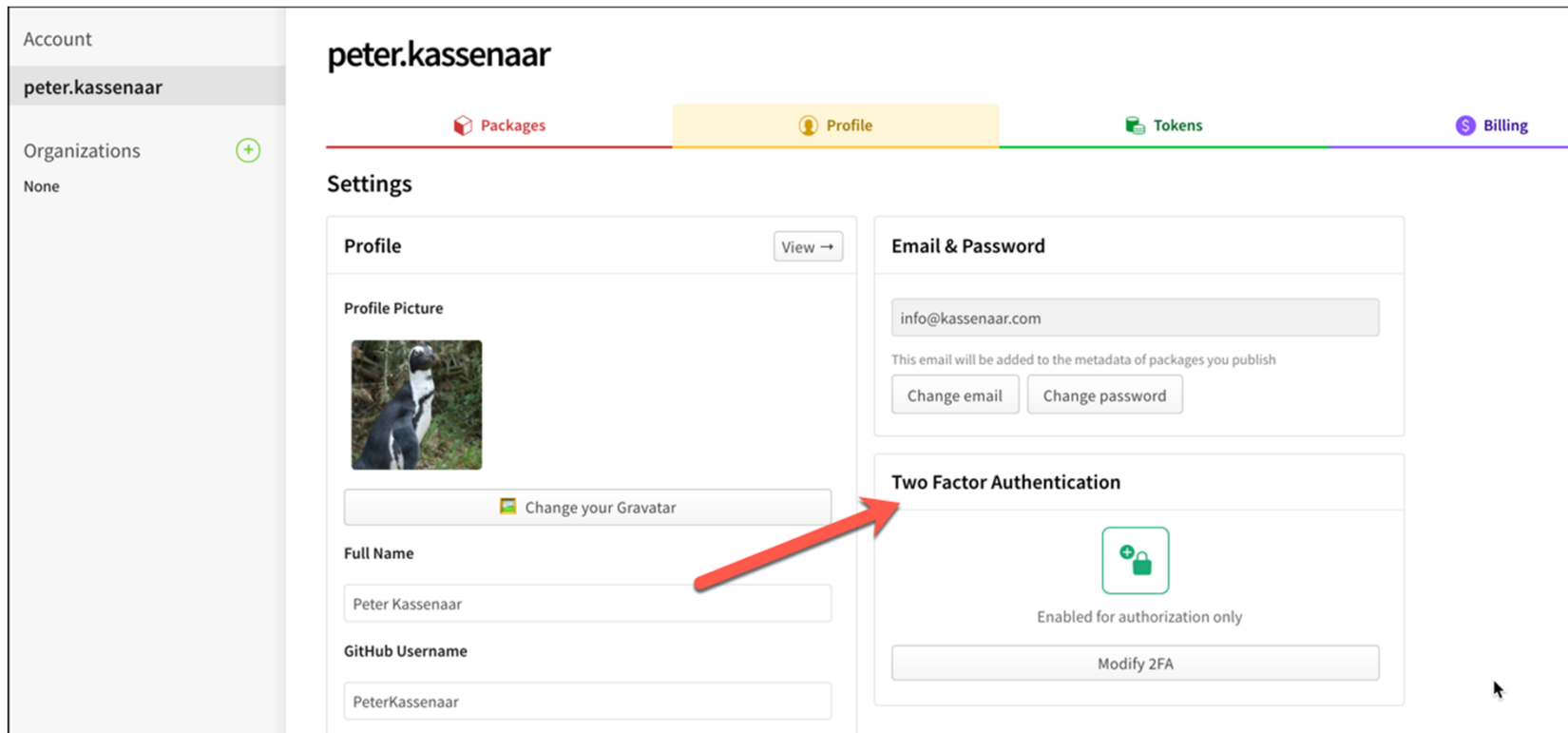


```
MacBook-Pro:~ PeterKassenaar$ npm logout
MacBook-Pro:~ PeterKassenaar$ npm login
Username: peter.kassenaar
Password:
Email: (this IS public) info@kassenaar.com
Enter one-time password from your authenticator app: 048803
Logged in as peter.kassenaar on https://registry.npmjs.org/.
MacBook-Pro:~ PeterKassenaar$
```

The screenshot shows a terminal window with the following text: `MacBook-Pro:~ PeterKassenaar$ npm logout`, `MacBook-Pro:~ PeterKassenaar$ npm login`, `Username: peter.kassenaar`, `Password:`, `Email: (this IS public) info@kassenaar.com`, `Enter one-time password from your authenticator app: 048803`, `Logged in as peter.kassenaar on https://registry.npmjs.org/.`, and `MacBook-Pro:~ PeterKassenaar$`. Two red arrows point from the top right towards the terminal: one points to the `npm login` command, and the other points to the one-time password `048803`.

Check/update 2FA


→ Go to your profile online and en-/disable 2FA



Am I logged in?

- `npm whoami`

```
MacBook-Pro:~ PeterKassenaar$ npm whoami  
peter.kassenaar  
MacBook-Pro:~ PeterKassenaar$
```

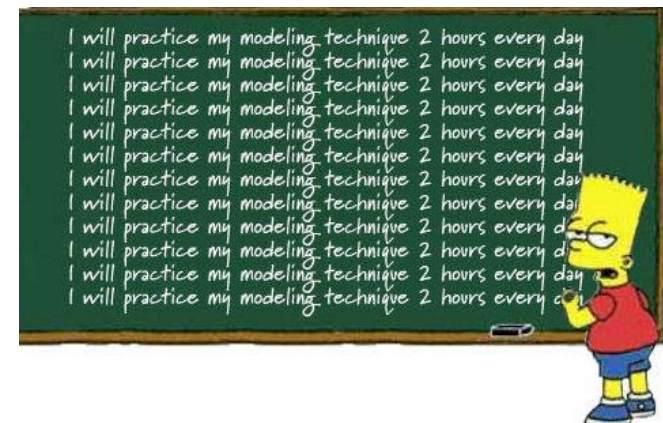


Logging out:

```
npm logout
```

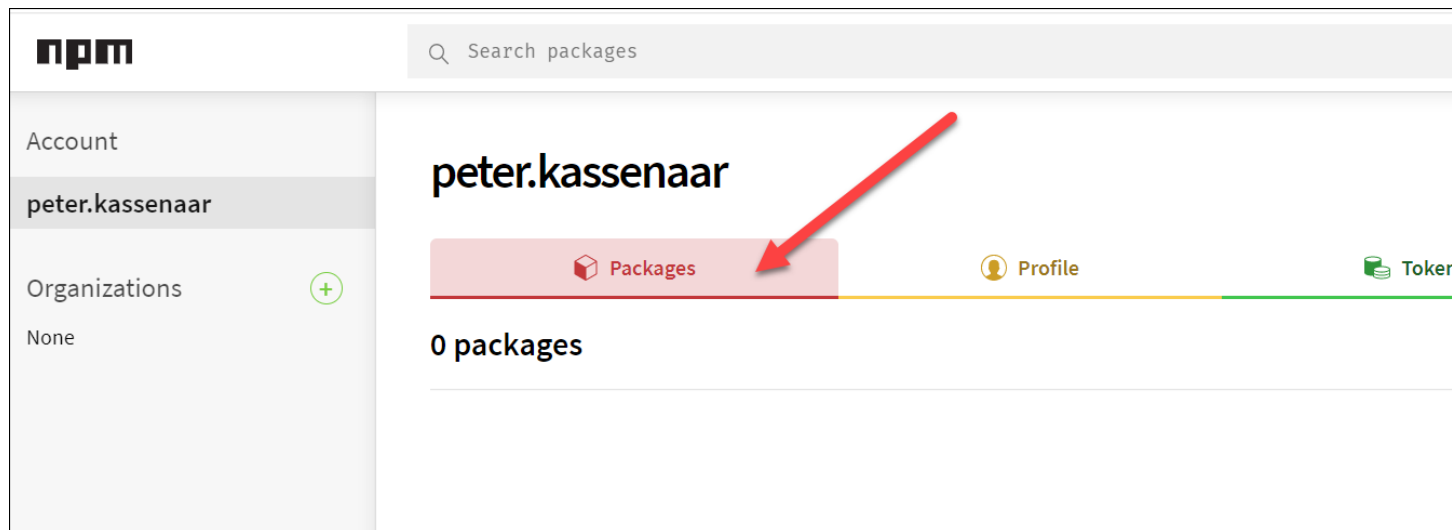
Mini Workshop

- Create an online account with npm
 - <https://www.npmjs.com/signup>
- OR (if you already have an account)
 - Check if you can log in/log out
 - Use `npm whoami` to check
 - Check your user/profile page online. Set up 2FA



Publishing a package

- Use your own package, created earlier
 - OR: use 100-simple-module
- We're going to publish this package to npm, so *anyone* can do an `npm install`



Step 1 – give it a name & version

- Recommended: create a @scoped/name
 - Edit `package.json`
 - Less/no chance of naming conflicts of your package with @scoping
- Give it a version number.
 - Must follow the SemVer rules `x.x.x`

```
{  
  "name": "@peter.kassenaar/pk-moment",  
  "version": "1.0.0",  
  ...  
}
```

Step 2 – add files to publish

- Add a `files: [...]` section to `package.json`,
 - Tell npm which files should be published
 - Not mandatory. But otherwise *all* files would be published,
 - Including editor config files like `.idea`, `.vscode`, etc.

```
{  
  "files": [  
    "*.js",  
    "*.json"  
  ],  
  ...  
}
```

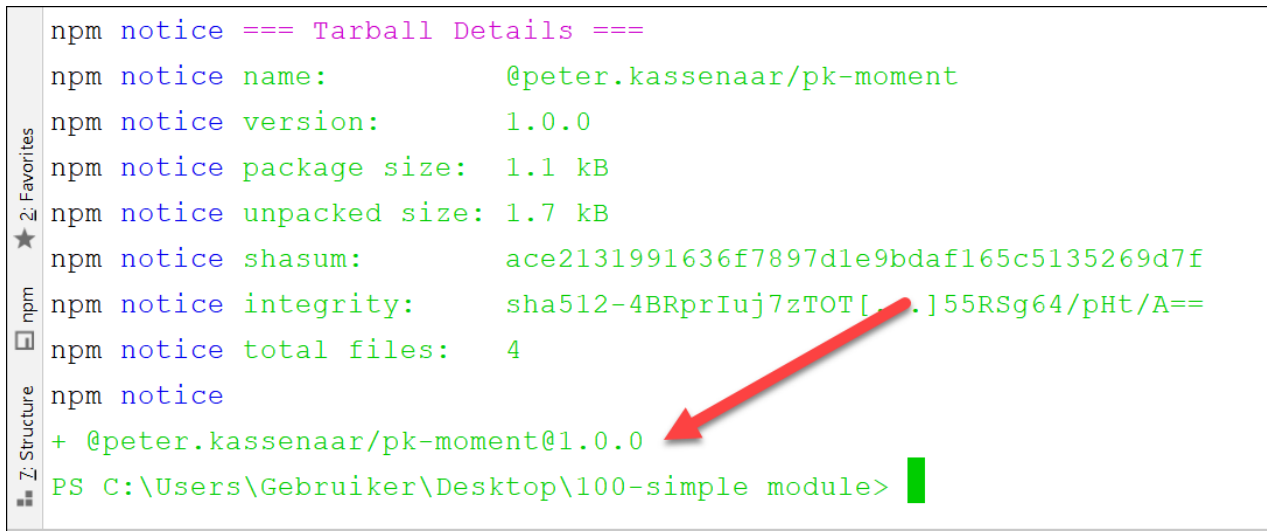
Step 3 – Try publishing

- `npm publish`
- Publish your created package
 - Error: '402 Payment Required (...) sign up for private packages'

```
npm notice name:      @peter.kassenaar/pk-moment
npm notice version:   1.0.0
npm notice package size: 1.1 kB
npm notice unpacked size: 1.7 kB
npm notice shasum:    ace2131991636f7897d1e9bdaf165c5135269d7f
npm notice integrity: sha512-4BRprIuj7zTOT[...]55RSg64/pHt/A==
npm notice total files: 4
npm notice
npm ERR! code E402
npm ERR! 402 Payment Required - PUT https://registry.npmjs.org/@peter.kassenaar%2fpk-moment - You must sign up for private packages
```


3a - Set publish access to public


- Normally, you can only have *private* repositories with a @scoped name
- However, if you explicitly set its access to public, you're fine.
- `npm publish --access=public`



```
npm notice === Tarball Details ===
npm notice name:          @peter.kassenaar/pk-moment
npm notice version:       1.0.0
npm notice package size:  1.1 kB
npm notice unpacked size: 1.7 kB
npm notice shasum:        ace2131991636f7897d1e9bdaf165c5135269d7f
npm notice integrity:     sha512-4BRprIuj7zTOT[.]55RSg64/pHt/A==
npm notice total files:   4
npm notice
+ @peter.kassenaar/pk-moment@1.0.0
PS C:\Users\Gebruiker\Desktop\100-simple module>
```





npm loves you

Successfully published @peter.kassenaar/pk-moment@1.0.0




npm Inc support <support@npmjs.com>

Aan Peter Kassenaar



08:30

 Er zijn extra regels verwijderd in dit bericht.

Hi peter.kassenaar!

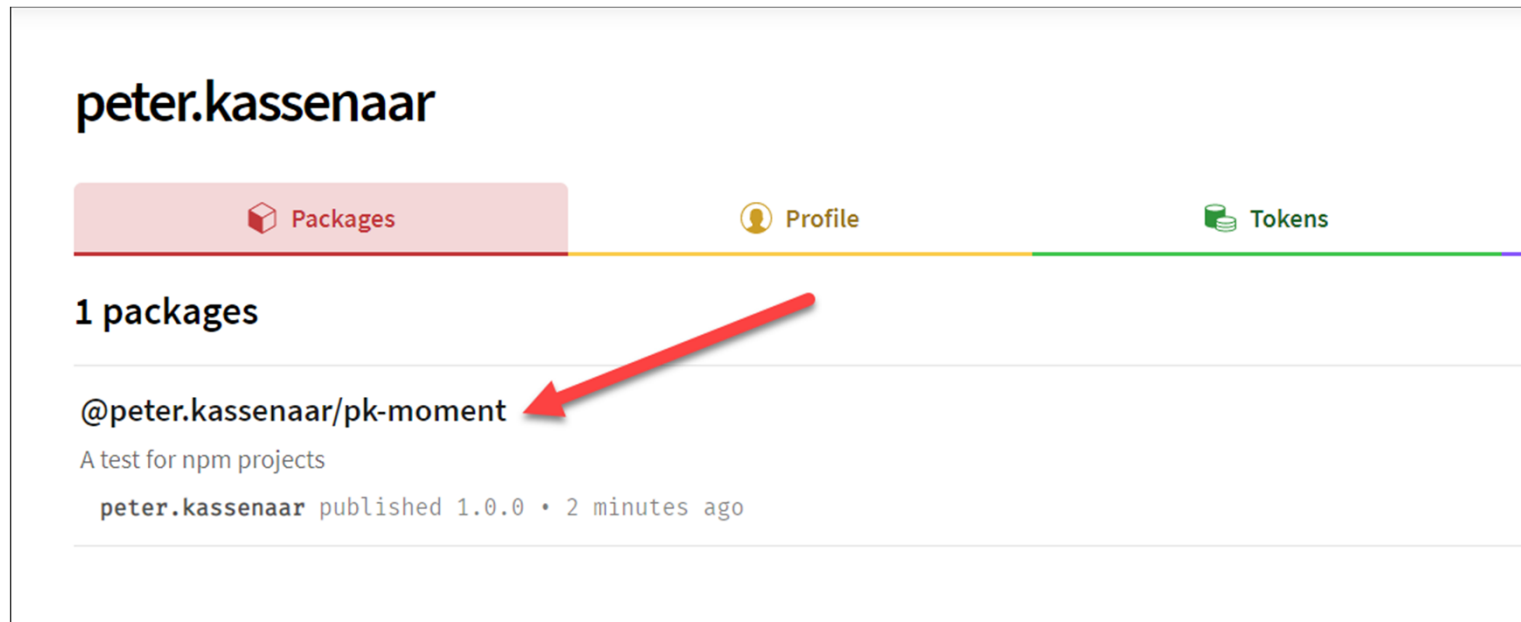
A new version of the package @peter.kassenaar/pk-moment (1.0.0) was published at 2019-09-04T06:29:45.339Z from 84.95.105.57. The checksum of this package was ace2131991636f7897d1e9bdaf165c5135269d7f.

If you have questions or security concerns, you can reply to this message or email support@npmjs.com.

npm loves you.

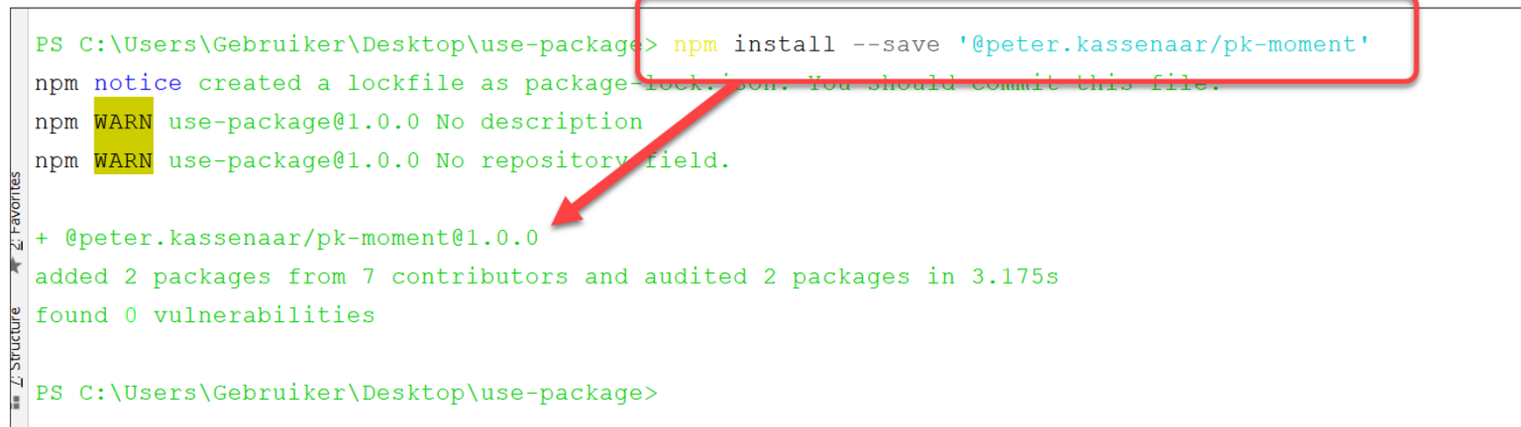
Result

- Check result online



Step 5 – Using your package

- Create a **new project**, using `npm init -y`
- Install your package,
 - `npm install --save @your-name/package-name`
 - On Windows: you need quotes around the name!

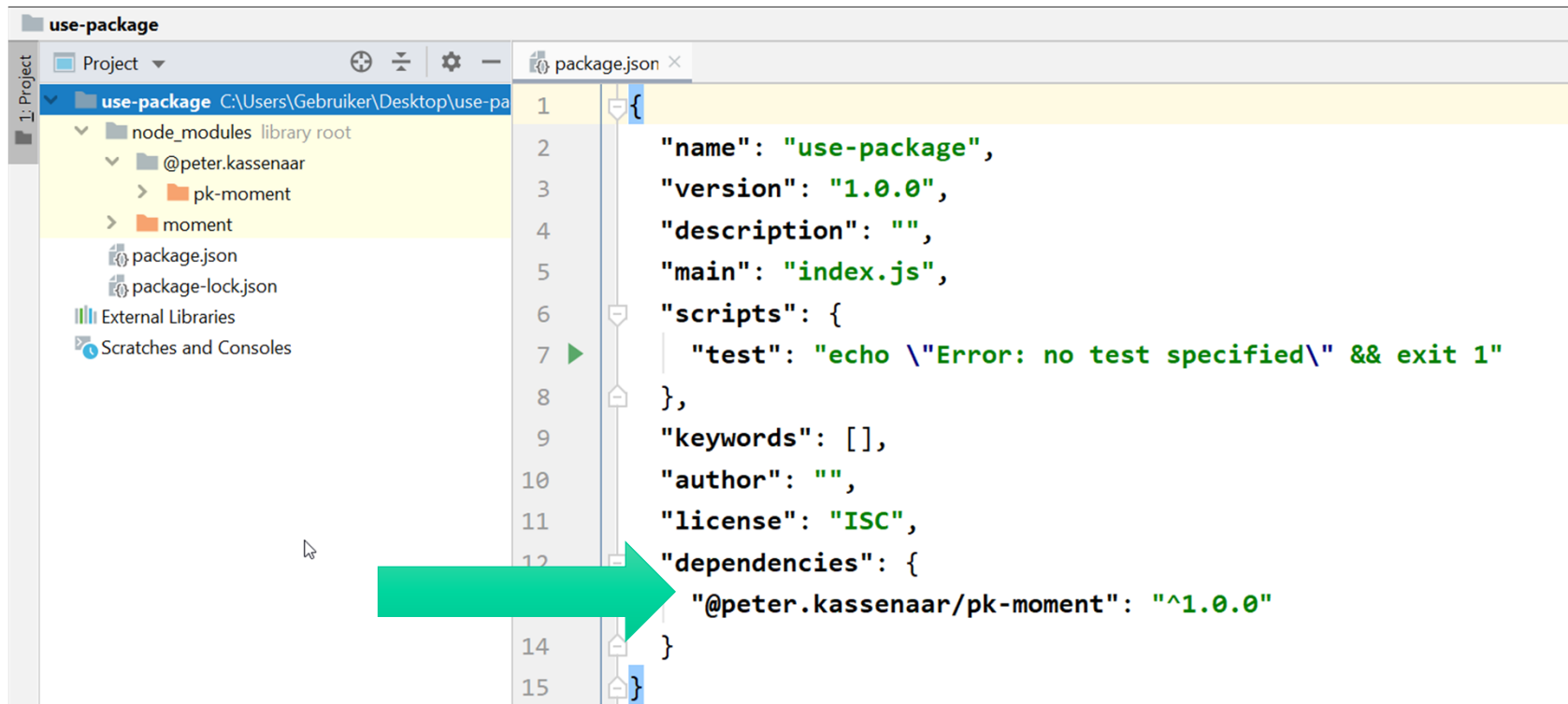


```
PS C:\Users\Gebruiker\Desktop\use-package> npm install --save '@peter.kassenaar/pk-moment'
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN use-package@1.0.0 No description
npm WARN use-package@1.0.0 No repository field.

+ @peter.kassenaar/pk-moment@1.0.0
added 2 packages from 7 contributors and audited 2 packages in 3.175s
found 0 vulnerabilities

PS C:\Users\Gebruiker\Desktop\use-package>
```

Result

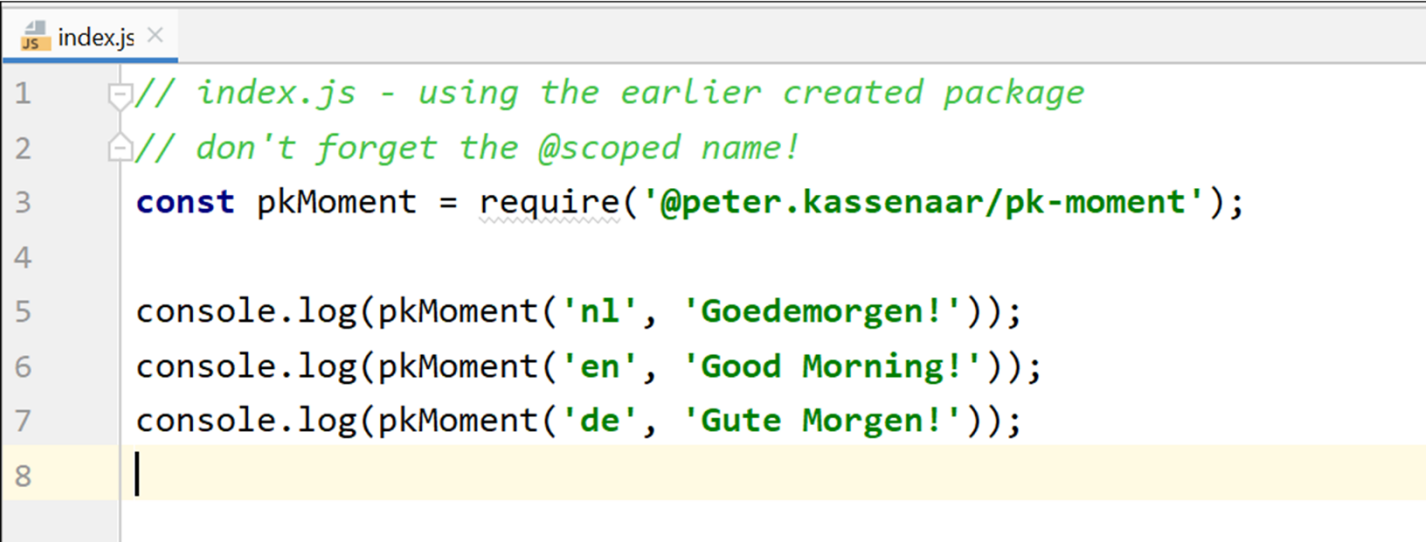


```
1 {
2   "name": "use-package",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "keywords": [],
10  "author": "",
11  "license": "ISC",
12  "dependencies": {
13    "@peter.kassenaar/pk-moment": "^1.0.0"
14  }
15 }
```

The screenshot shows a code editor with a project named 'use-package'. The left sidebar displays the project structure, including a 'node_modules' directory with subdirectories '@peter.kassenaar' and 'pk-moment'. The main editor area shows the 'package.json' file. A green arrow points to the dependency entry for '@peter.kassenaar/pk-moment' in the 'dependencies' object.

Step 6 – adding the package

- Write a new `index.js` file in the new project
 - Don't bother about `./node_modules`!
 - Load your package and use it as normal


A screenshot of a code editor window with a tab labeled 'index.js'. The editor contains the following JavaScript code:

```
1 // index.js - using the earlier created package
2 // don't forget the @scoped name!
3 const pkMoment = require('@peter.kassenaar/pk-moment');
4
5 console.log(pkMoment('nl', 'Goedemorgen!'));
6 console.log(pkMoment('en', 'Good Morning!'));
7 console.log(pkMoment('de', 'Gute Morgen!'));
8 |
```

The code is color-coded: comments are green, the `const` keyword is blue, and the `require` function is underlined. The editor has a light gray background and a yellow highlight on the line containing the cursor.

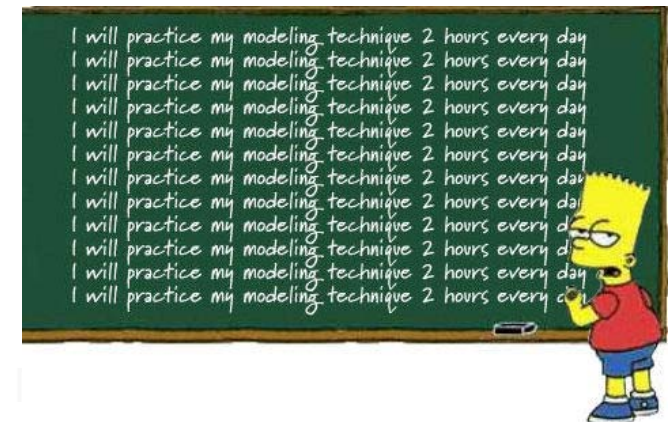
Output

```
[nodemon] 1.19.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] starting `node .\index.js`
4 sep. 2019 08:42, >> Goedemorgen!
Sep 4, 2019 8:42 AM, >> Good Morning!
4. Sep. 2019 08:42, >> Gute Morgen!
[nodemon] clean exit - waiting for changes before restart
```



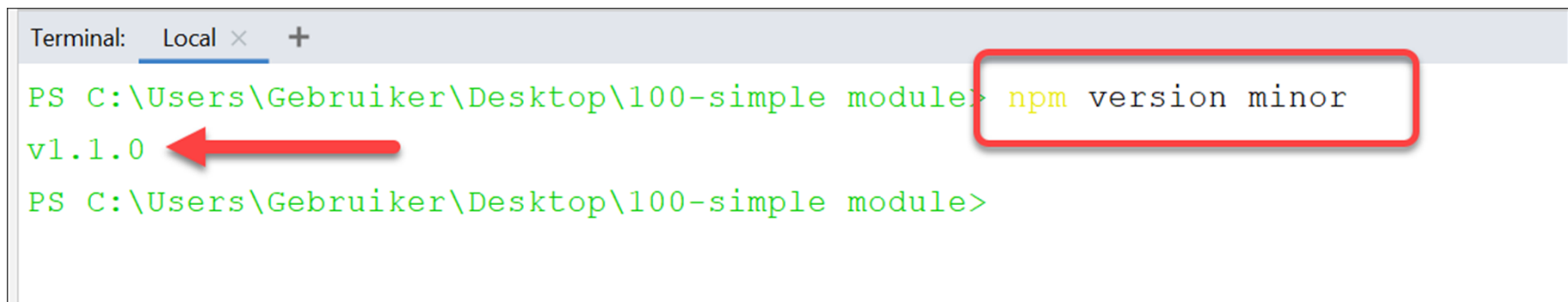
Workshop

- Create and publish your own package!
 - or – use `./100-simple-module` as an example
- DO use your own scoped name
- Publish with the `access=public` modifier



Updating your package

- Open original project
- Write new code, refactor, add functionality, etc.
- `npm version major | minor | patch`
- Updates the version number in `package.json`
- `npm publish` again.

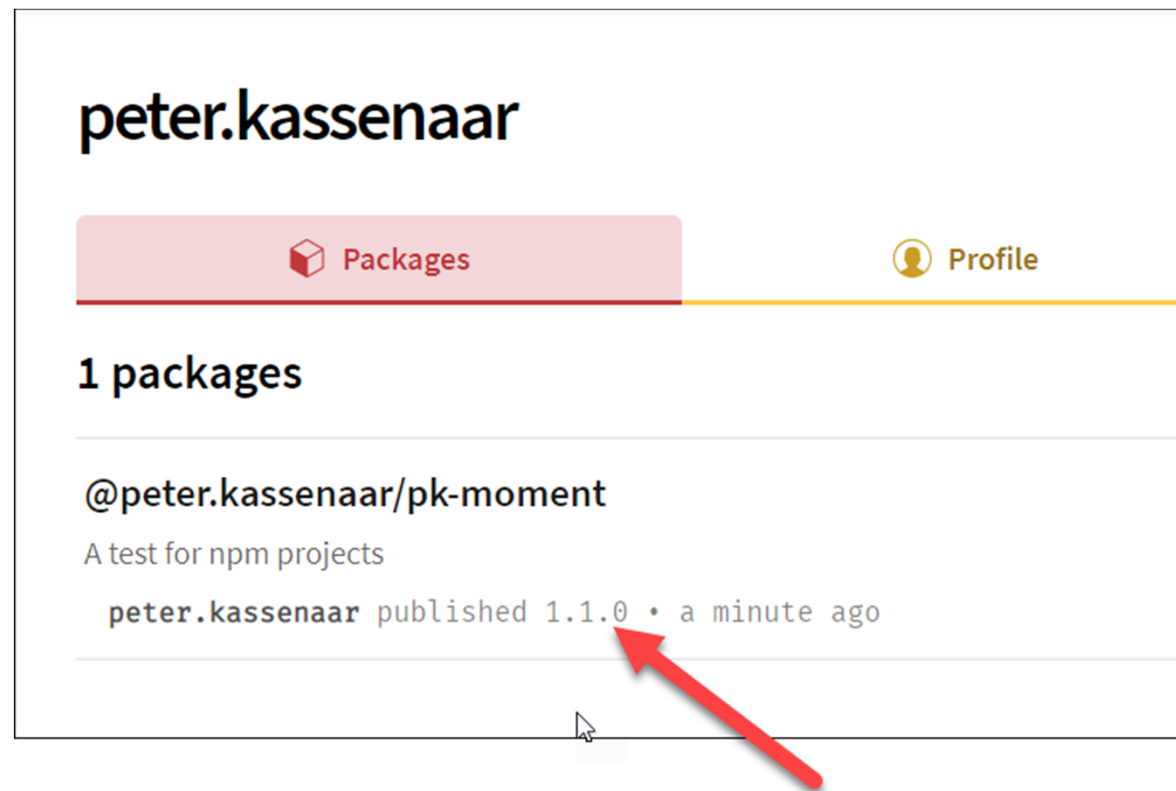


A screenshot of a Windows Command Prompt terminal window. The title bar shows 'Terminal: Local x +'. The command prompt shows the current directory as 'C:\Users\Gebruiker\Desktop\100-simple module'. The command `npm version minor` has been entered and executed. The output is `v1.1.0`, which is highlighted by a red arrow pointing to it from the left. The command `npm version minor` is also enclosed in a red rectangular box.

```
Terminal: Local x +
PS C:\Users\Gebruiker\Desktop\100-simple module> npm version minor
v1.1.0
PS C:\Users\Gebruiker\Desktop\100-simple module>
```

Re-publishing a new version

- When republishing a package, no need to add the `access=public` modifier



Check package online

- `https://www.npmjs.com/package/@scope.name/module-name`

@peter.kassenaar/pk-moment
1.1.0 • Public • Published a minute ago

Readme Admin 1 Dependencies 0 Dependents **2 Versions**

Tip: Click on a version number to view a previous version's package page

Current Tags

1.1.0	latest
-------	--------

Version History

1.1.0	a minute ago
1.0.0	20 minutes ago

install
> npm i @peter.kassenaar/pk-moment

version	license
1.1.0	ISC

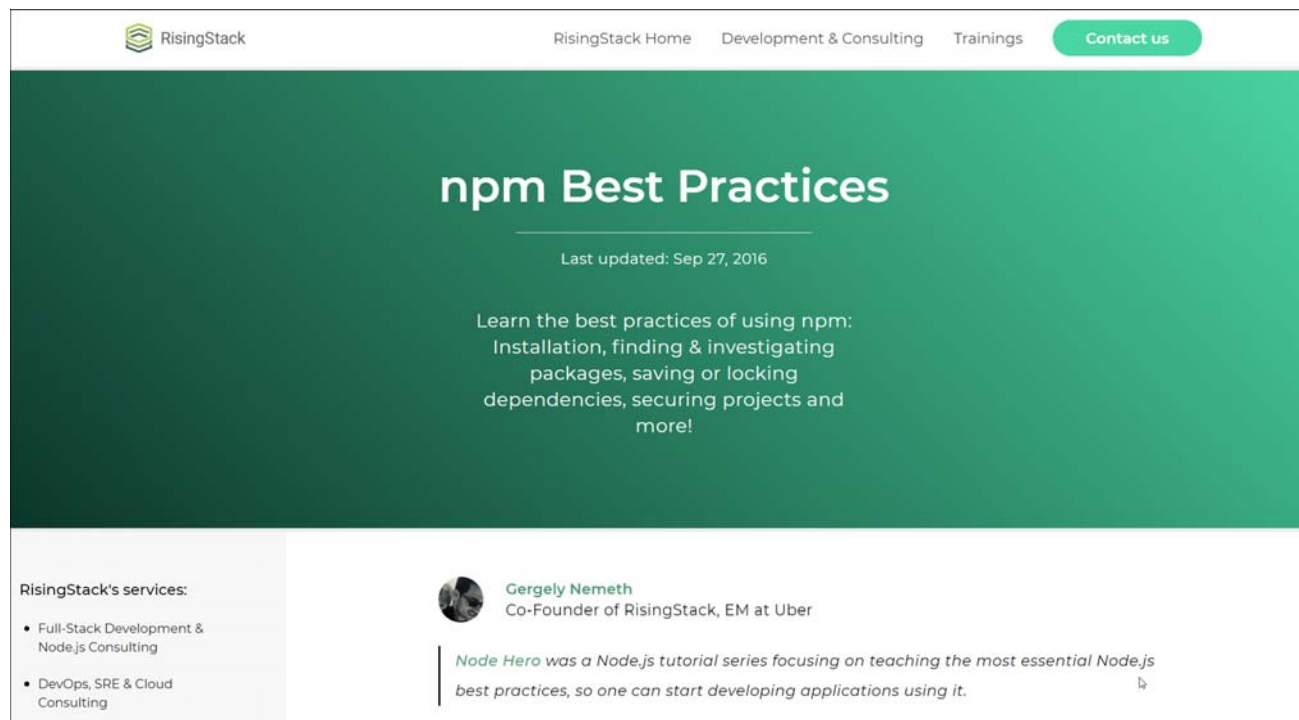
last publish
a minute ago

collaborators

<https://www.npmjs.com/package/@peter.kassenaar/pk-moment>

Best Practices

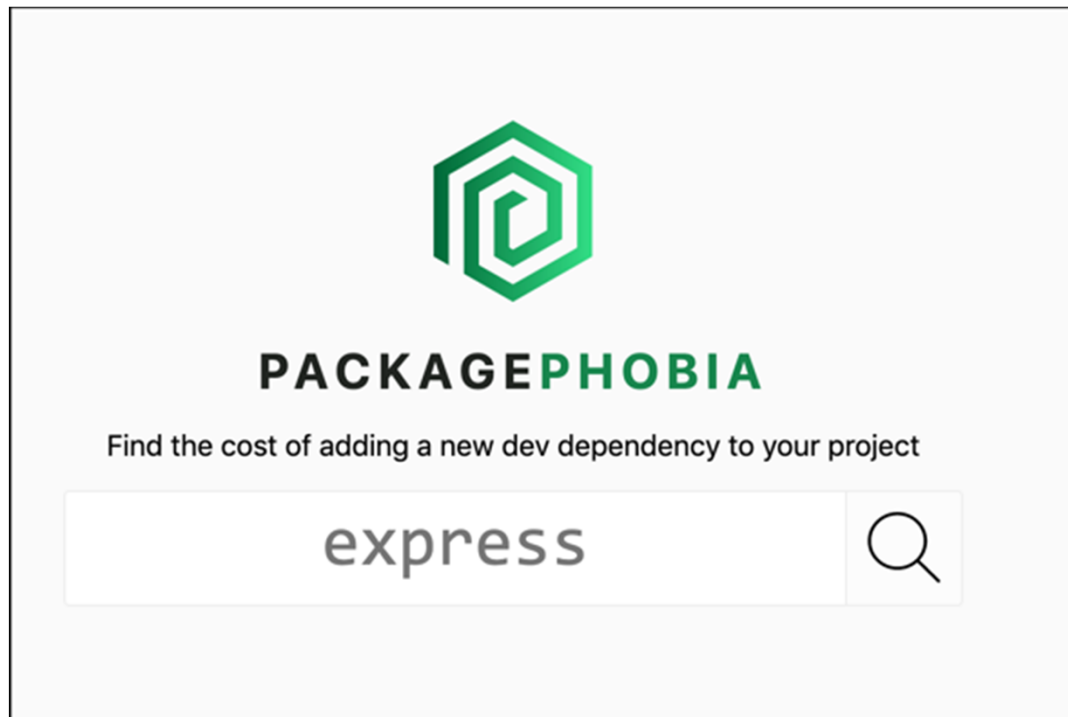
- Add a `README.md` file to the project
- Analyze and update keywords in the `package.json` file



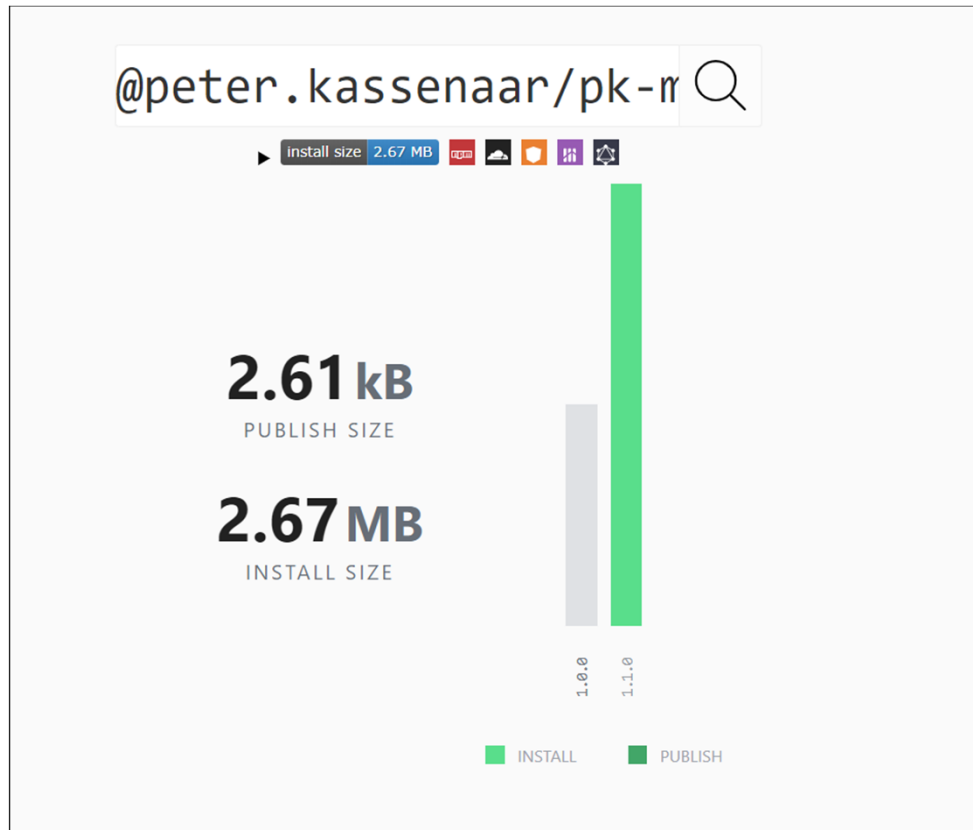
<https://blog.risingstack.com/nodejs-at-scale-npm-best-practices/>

How big is your package?

- <https://packagephobia.now.sh/>



Results for our package



Question: why is the `install` size so much bigger than the `publish` size?

Answer: because of the dependencies

Deleting your package online

- BAD practice!
 - Because others might depend on your package
- But it is possible, within 72hrs of publication
- <https://docs.npmjs.com/cli/unpublish.html>

CLI documentation > CLI commands

npm-unpublish

Remove a package from the registry

SYNOPSIS

```
npm unpublish [<@scope>/]<pkg>[@<version>]
```

WARNING

It is generally considered bad behavior to remove versions of a library that others are depending on!

Consider using the **deprecate** command instead, if your intent is to encourage users to upgrade.

There is plenty of room on the registry.

Use the --force key

```
Opdrachtprompt
Microsoft Windows [Version 10.0.18362.295]
(c) 2019 Microsoft Corporation. Alle rechten voorbehouden.

C:\Users\Gebruiker>npm unpublish @peter.kassenaar/pk-moment
npm ERR! Refusing to delete entire project.
npm ERR! Run with --force to do this.
npm ERR! undefined

C:\Users\Gebruiker>
```

```
C:\Users\Gebruiker>npm unpublish @peter.kassenaar/pk-moment --force
npm WARN using --force I sure hope you know what you are doing.
-@peter.kassenaar/pk-moment

C:\Users\Gebruiker>
```


TL; DR

- *“An npm module **only** requires a package.json file with **name** and **version** properties.”*

<https://medium.com/free-code-camp/how-to-make-a-beautiful-tiny-npm-package-and-publish-it-2881d4307f78>

Workshop

- 1. Make some adjustments to your package
- Bump the version number, using `npm version` (major, minor or patch)
- Republish the new version and check online
- Optional – update/use the new version in your –other- project
- 2. Remove the package online
 - `npm unpublish`

