Semantic Versioniong

2	3	0
• Major Release • Breaking Change	• Minor Release • New Features added	• Patch Release • Bug Fixes

Full Stack Web Development

https://semver.org/

Semantic Versioning 2.0.0

Summary

Given a version number MAJOR.MINOR.PATCH, increment the:

1. MAJOR version when you make incompatible API changes,
2. MINOR version when you add functionality in a backwards-compatible manner, and
3. PATCH version when you make backwards-compatible bug fixes.

Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.

Semantic Versioning of Packages

- Semantic versioning is a standard node projects use to communicate what kinds of changes are in a new release.
- Sometimes those changes will break the code that depends on the package.

```
"dependencies": {
  "boom": "^7.3.0",
  "dotenv": "^6.2.0",
  "handlebars": "^4.0.12",
  "hapi": "^18.0.0",
  "hapi-auth-cookie": "^9.1.0",
  "inert": "^5.1.2",
  "joi": "^14.3.1",
  "mongoose": "^5.4.7",
  "vision": "^5.4.4"
},
```

- 3-component system in the format of x.y.z where:
  - x stands for a major version
  - y stands for a minor version
  - z stands for a patch
- Major.Minor.Patch.

^ Symbol

- The caret ^ range specifier permits automatic upgrades to minor version increments of a package
- So if ‘npm install’ is invoked, the actual version downloaded and installed may be more recent that the one enumerated in package.json
- For caret ranges, only major version must match. Any minor or patch version greater than or equal to the minimum is valid.

```
"dependencies": {
  "boom": "^7.3.0",
  "dotenv": "^6.2.0",
  "handlebars": "^4.0.12",
  "hapi": "^18.0.0",
  "hapi-auth-cookie": "^9.1.0",
  "inert": "^5.1.2",
  "joi": "^14.3.1",
  "mongoose": "^5.4.7",
  "vision": "^5.4.4"
},
```

- Example
  - ^1.2.3 permits versions from 1.2.3 all the way up to, but not including, the next major version, 2.0.0.

Updating all Dependencies

npm-check-updates public

Find newer versions of dependencies than what your package.json or bower.json allows

npm package 2.8.5 | build passing | dependencies up-to-date

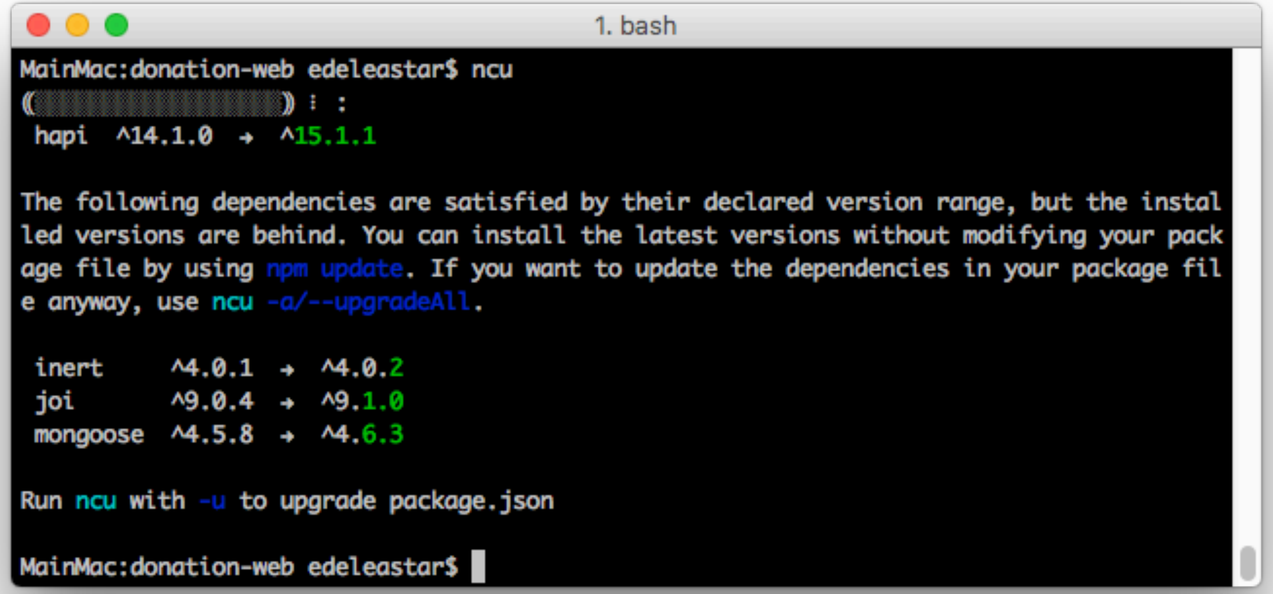
npm-check-updates is a command-line tool that allows you to upgrade your package.json or bower.json dependencies to the latest versions, regardless of existing version constraints.

```
$ npm install npm-check-updates -g
```

Report on Dependency Status (no change)

```
$ ncu
```

```
"dependencies": {
  "handlebars": "^4.0.5",
  "hapi": "^14.1.0",
  "hapi-auth-cookie": "^6.1.1",
  "inert": "^4.0.1",
  "joi": "^9.0.4",
  "mongoose": "^4.5.8",
  "vision": "^4.1.0"
}
```



Force upgrade all dependencies

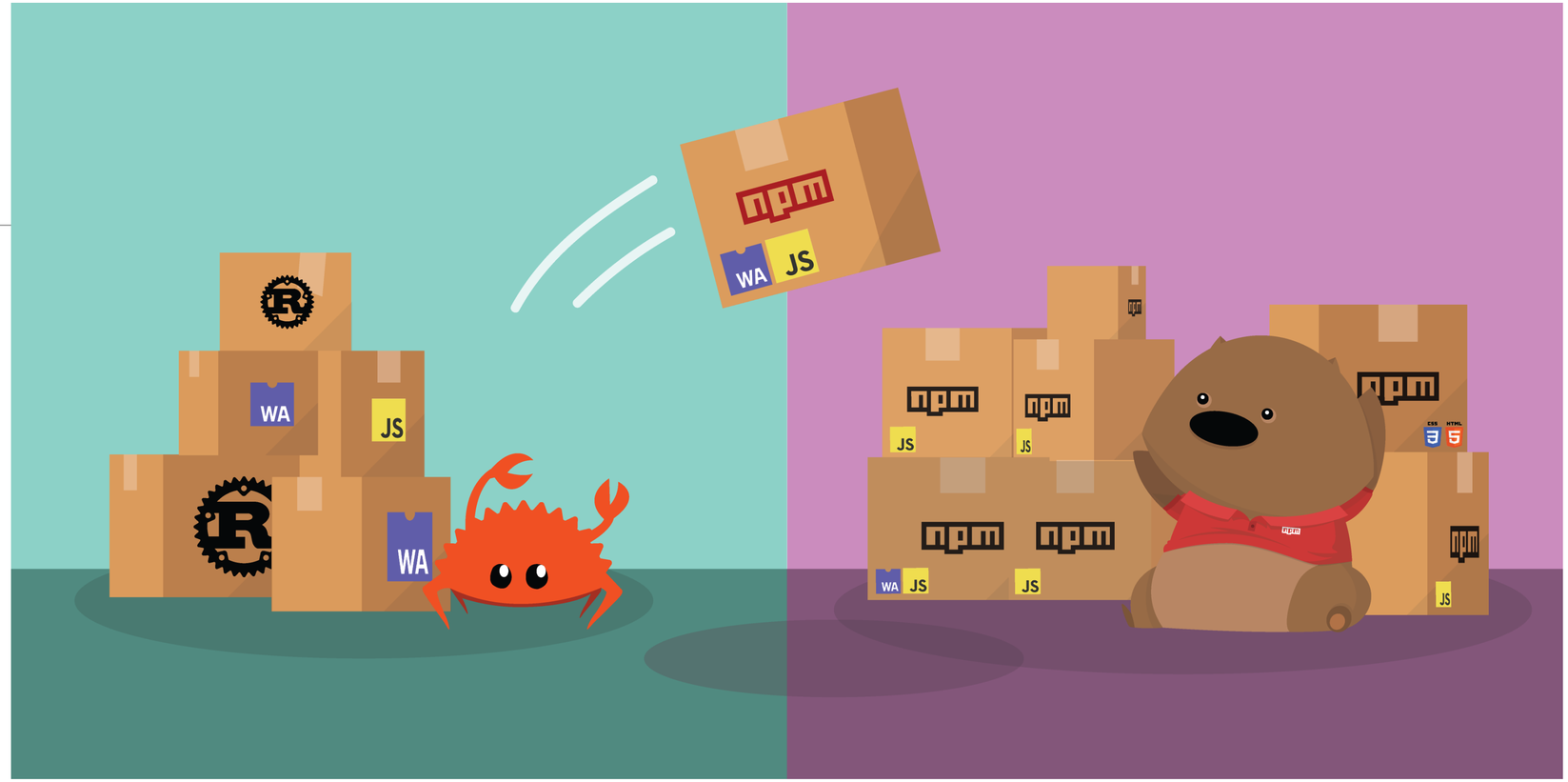
```
$ ncu -u
```

```
"dependencies": {
  "handlebars": "^4.0.5",
  "hapi": "^14.1.0",
  "hapi-auth-cookie": "^6.1.1",
  "inert": "^4.0.1",
  "joi": "^9.0.4",
  "mongoose": "^4.5.8",
  "vision": "^4.1.0"
}
```

```
"dependencies": {
  "handlebars": "^4.0.5",
  "hapi": "^15.1.1",
  "hapi-auth-cookie": "^6.1.1",
  "inert": "^4.0.1",
  "joi": "^9.0.4",
  "mongoose": "^4.5.8",
  "vision": "^4.1.0"
}
```

Npm Scopes

- All npm packages have a name.
- Some package names also have a scope
- Scopes are a way of grouping related packages together




```
@somescope/somepackagename
```

node_modules/

`@somescope/somepackagename`

- Scoped packages are installed to a sub-folder of the regular installation folder,
  - e.g. if your other packages are installed in node_modules/packagename, scoped modules will be installed in node_modules/@myorg/packagename.
- The scope folder (@myorg) is simply the name of the scope preceded by an @ symbol, and can contain any number of scoped packages.

Installing Scoped Packages

- A scoped package is installed by referencing it by name, preceded by an @ symbol, in npm install:

```
npm install @myorg/mypackage
```

```
"dependencies": {
  "@myorg/mypackage": "^1.3.0"
}
```

hapijs / hapi

Code | Issues 9 | Pull requests 1 | Projects 0 | Wi

Releases | Tags

Tags

v18.1.0
6 days ago 5ced8ae zip tar.gz

v17.8.4
6 days ago 054482c zip tar.gz

v18.0.1
10 days ago 4b59b3f zip tar.gz

v17.8.3
10 days ago

v17.8.2
10 days ago

v18.0.0
23 days ago

Getting Started

This tutorial is compatible with hapi v17 and newer

Overview

This tutorial will show how to set up a basic hapi server that displays "Hello World!" in your browser.

Installing hapi

Create a new directory `myproject`, and from there:

- Run: `cd myproject`, this goes into the created project folder.
- Run: `npm init` and follow the prompts. This will generate a package.json file for you.
- Run: `npm install @hapi/hapi`, this will install the latest version of hapi as a dependency in your package.json.

A rich framework for buildi

hapi enables developers to focus on writing reusable application logic instead of spending time building infrastructure.

```
$ npm install hapi
```

Current version: 18.1.0 • Latest update: 4 days ago • Downloads last month: 894,258

hapi.js versions

How one programmer broke the internet by deleting a tiny piece of code

leftpad.js package.json

```
1  module.exports = leftpad;
2  function leftpad (str, len, ch) {
3    str = String(str);
4    var i = -1;
5    if (!ch && ch !== 0) ch = ' ';
6    len = len - str.length;
7    while (++i < len) {
8      str = ch + str;
9    }
10   return str;
11 }
12
13
14
15
16
17
18
19
20
```

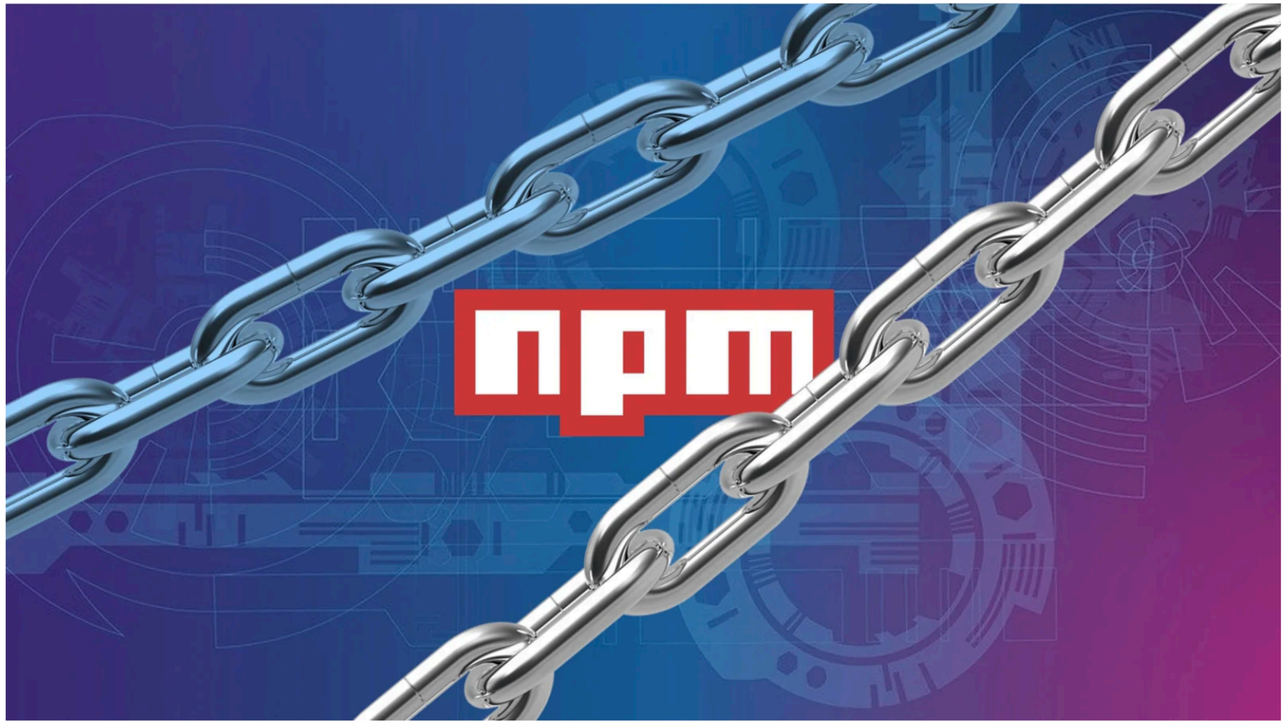
The left pad incident

"A man in Oakland, California, disrupted web development around the world last week by deleting 11 lines of code."

Dev corrupts NPM libs 'colors' and 'faker' breaking thousands of apps

By Ax Sharma

January 9, 2022 09:17 AM 32



Users of popular open-source libraries 'colors' and 'faker' were left stunned after they saw their applications, using these libraries, printing gibberish data and breaking.

Some surmised if the NPM libraries had been compromised, but it turns out there's much more to the story.

The developer of these libraries intentionally introduced an infinite loop that bricked **thousands of projects** that depend on 'colors' and 'faker.'

Semantic Versioniong

2	3	0
• Major Release • Breaking Change	• Minor Release • New Features added	• Patch Release • Bug Fixes

Full Stack Web Development