

SUPERCHARGING THE TERMINAL WITH NODEJS

✈️ takeoff for JS developers.



HENDRIK

JavaScript enthusiast

Developer for fun 🎉

```
hewa@ham-hewa MINGW64 ~/dev
$ cd http-status/

hewa@ham-hewa MINGW64 ~/dev/http-status (master)
$ ls
codes.json  index.js*  node_modules/  package.json  README.md

hewa@ham-hewa MINGW64 ~/dev/http-status (master)
$ cat package.json | pjs -f "match(/version/)" -m "split('\n')[3]"
1.0.0

hewa@ham-hewa MINGW64 ~/dev/http-status (master)
$
```

•Terminal

Development needs it, we love it, we hate it, we use it!

CLIs with NodeJS

- ▶ You can build a CLI in Node
- ▶ JS developers can read that



MAKE EVERYDAY TASKS BETTER

Make everyday tasks better

- ▶ Utility:
 - ▶ Vtop – An amazing version of top
 - ▶ Speed-test – Test you internet speed
 - ▶ Brightness-cli – Change your screen brightness
 - ▶ Alder – Tree with colors

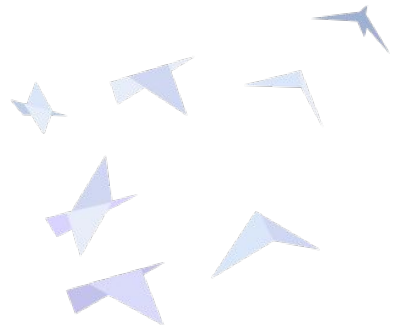


Tree using alder

```
hewa@ham-hewa MINGW64 ~/dev/http-status (master)
$ alder
```

```
├── codes.json
├── index.js
├── package.json
└── README.md
```

```
1 directories, 5 files (16.7 kB)
```



Make everyday tasks better

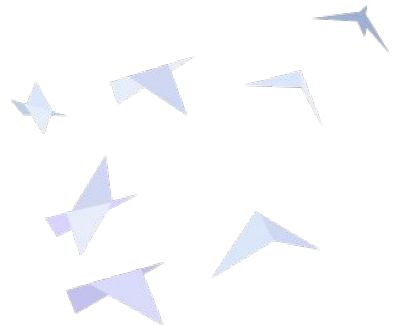
- Jsome – Nice JSON in your terminal
- Lessmd – Colored Markdown in the terminal
- Hget – Grab a text version of a website
- Pjs – Filter using JavaScript



Filter output with pjs

```
hewa@ham-hewa MINGW64 ~/dev/http-status (master)
$ jsome package.json
{
  name: "@hoverbaum/http-code",
  version: "1.0.0",
  description: "A super simple CLI to look up the meaning of http status codes.",
  main: "index.js",
  author: "Hendrik Wallbaum <mail@hendrikwallbaum.de>",
  dependencies: {
    commander: "^2.14.1"
  },
  bin: "index.js"
}

hewa@ham-hewa MINGW64 ~/dev/http-status (master)
$ cat package.json | pjs -f "match(/version/)" -m "split('\\')[3]"
1.0.0
```



A blurred photograph of an airport terminal. In the foreground, a dark rolling suitcase stands upright. The background shows a busy terminal with people walking, sitting at tables, and standing near information screens. The image has a blue-to-orange gradient overlay.

BEYOND THE TERMINAL

Beyond the terminal

- ▶ Diff2html – See git diff in browser
- ▶ Live-server – Preview websites in your browser
- ▶ Bcat – Cat to browser
- ▶ Ttystudio – Record terminal to gif
- ▶ Pagers – Screenshots of websites
- ▶ Pen – Markdown preview in browser



DO FUN THINGS



Do fun things

- Emoj – Find relevant emojis
- Normit – Translate text in your terminal
- Weather-cli – Up to date weather in on command
- Movie – Get information about a movie
- Vaca – Generate a random cow



Do fun things

```
hewa@ham-hewa MINGW64 ~/dev/http-status (master)
$ vaca

      ( _ )
      (oo)
      \V/
-----
  /  |  |  |  |
 /  |  |  |  |
*  |  |  |  |
  |  |  |  |
  ^ ^  ^ ^
Stretch Cow
```





AMAZING POTENTIAL



CASH



•Node based terminal

- Full featured Unix like terminal
- Compatible with all platforms
- Just use the commands


```
5. slap (node)
3,5 (101) UTF-8 Help: f2
1 | [Screenshot] (https://raw.githubusercontent.com/...-editor/
2 | \n
3 | slap: wave: · [! [Build Status] (https://img.shields.io/travis/sla
4 | =====\n
5 | \n
6 | slap is a Sublime-like terminal-based text editor that strives
7 | from the terminal easier. It has:\n
8 | \n
9 | * first-class mouse support (even over an SSH connection)\n
10 | * a Sublime-like file sidebar\n
11 | * double-click to select word, highlight other occurrences\n
12 | * configurable Sublime-like [keybindings] (....ini#L51) [*] (#so
13 | * copying/pasting with OS clipboard support\n
14 | * infinite undo/redo\n
15 | * syntax highlighting for [100+ languages] (https://github.com/
16 | * bracket matching\n
17 | * autoindentation\n
18 | * heavily customizable via [plugins] (#plugins)\n
19 | * ... many other features that will make you leave nano, vim,
20 | \n
21 | Installation\n
22 | _____\n
23 | \n
```

•Slap

- Node based editor in your terminal
- Mouse support
- Sidebar with file tree
- Syntax highlighting, plugins



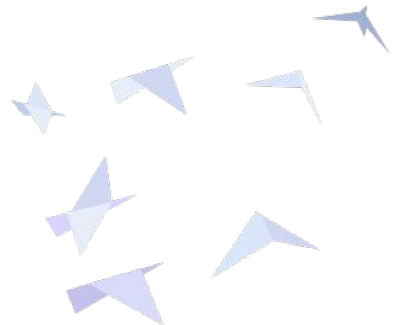
LETS BUILD

HTTP-Codes

```
hewa@ham-hewa MINGW64 ~/dev/http-status (master)
$ http-code 404
```

```
404: Not Found
```

The server can not find requested resource. In the browser, this means the URL is not recognized. In an API, this can also mean that the endpoint is valid but the resource itself does not exist. Servers may also send this response instead of 403 to hide the existence of a resource from an unauthorized client. This response code is probably the most famous one due to its frequent occurrence on the web.



```
hewa@ham-hewa MINGW64 ~/dev/http-status (master)
$ alder --exclude=commander
```

```
.
├── codes.json
├── index.js
├── node_modules
├── package.json
└── README.md
3 directories, 5 files (16.6 kB)
```

HTTP-Codes

- Display information about a given HTTP status code.

Idea:

- JSON file with information
- Handle user input and find right information to display
- Otherwise show a message

```
JS index.js x
1  #!/usr/bin/env node
2
3  const cli = require('commander')
4  let statusCode = null
5
6  cli
7    .version(require('./package.json').version)
8    .arguments('<status code>')
9    .action(code => {
10     | statusCode = code
11     | })
12    .parse(process.argv)
13
14  if (!statusCode) {
15    console.log('No status Code supplied. Please specify a status code you are looking for.')
16    process.exit(1)
17  }
18
19  const codes = require('./codes.json')
20  statusCode = parseInt(statusCode)
21
22  const code = codes.find(code => code.number === statusCode)
23  if (!code) {
24    console.log('Could not find the code you are looking for.\nIf you think it should exist please consider opening an Issue.')
25    process.exit(1)
26  }
27
28  console.log(`
29  ${code.number}: ${code.name}
30
31  ${code.description}
32  `)
33
```

•Dive in

- Commander for CLI
- Define usage
- Parse arguments
- Check input
- Provide information

```
1  #! /usr/bin/env node
2
3  const cli = require('commander')
4  let statusCode = null
5
6  cli
7    .version(require('./package.json').version)
8    .arguments('<status code>')
9    .action(code => {
10     |   statusCode = code
11     | })
12   .parse(process.argv)
13
14  if (!statusCode) {
15     | console.log('No status Code supplied. Please specify a status code you are looking for.')
16     | process.exit(1)
17   }
18
19  const codes = require('./codes.json')
20  statusCode = parseInt(statusCode)
21
22  const code = codes.find(code => code.number === statusCode)
23  if (!code) {
24     | console.log('Could not find the code you are looking for.\nIf you think it should exist please consider opening an Issue.')
25     | process.exit(1)
26   }
27
28  console.log(`
29  ${code.number}: ${code.name}
30
31  ${code.description}
32  `)
33
```

Further reading

- Lists of amazing tools:
 - <https://github.com/sindresorhus/awesome-nodejs#command-line-apps>
 - <https://github.com/agarrharr/awesome-cli-apps>
- Winston – <https://github.com/tj/commander.js>
- Vorpals – <https://github.com/dthree/vorpals>
- Pictures from: <https://unsplash.com/>



HENDRIK WALLBAUM

hendrik.wallbaum@netlight.com
@hoverbaum

netlight.com

