

# nodemon

nodemon is a tool that helps develop Node.js based applications by automatically restarting the node application when file changes in the directory are detected.

nodemon does **not** require *any* additional changes to your code or method of development. nodemon is a replacement wrapper for node. To use nodemon, replace the word node on the command line when executing your script.

[NPM version](https://npmjs.org/package/nodemon) [Backers on Open Collective](#1pxezwc) [Sponsors on Open Collective](#49x2ik5)

# Installation

Either through cloning with git or by using [npm](http://npmjs.org) (the recommended way):

npm install -g nodemon # or using yarn: yarn global add nodemon

And nodemon will be installed globally to your system path.

You can also install nodemon as a development dependency:

npm install --save-dev nodemon # or using yarn: yarn add nodemon -D

With a local installation, nodemon will not be available in your system path or you can't use it directly from the command line. Instead, the local installation of nodemon can be run by calling it from within an npm script (such as npm start) or using npx nodemon.

# Usage

nodemon wraps your application, so you can pass all the arguments you would normally pass to your app:

nodemon [your node app]

For CLI options, use the -h (or --help) argument:

nodemon -h

Using nodemon is simple, if my application accepted a host and port as the arguments, I would start it as so:

nodemon ./server.js localhost 8080

Any output from this script is prefixed with [nodemon], otherwise all output from your application, errors included, will be echoed out as expected.

You can also pass the inspect flag to node through the command line as you would normally:

nodemon --inspect ./server.js 80

If you have a package.json file for your app, you can omit the main script entirely and nodemon will read the package.json for the main property and use that value as the app ([ref](https://github.com/remy/nodemon/issues/14)).

nodemon will also search for the scripts.start property in package.json (as of nodemon 1.1.x).

Also check out the [FAQ](https://github.com/remy/nodemon/blob/master/faq.md) or [issues](https://github.com/remy/nodemon/issues) for nodemon.

## Automatic re-running

nodemon was originally written to restart hanging processes such as web servers, but now supports apps that cleanly exit. If your script exits cleanly, nodemon will continue to monitor the directory (or directories) and restart the script if there are any changes.

## Manual restarting

Whilst nodemon is running, if you need to manually restart your application, instead of stopping and restart nodemon, you can type rs with a carriage return, and nodemon will restart your process.

## Config files

nodemon supports local and global configuration files. These are usually named nodemon.json and can be located in the current working directory or in your home directory. An alternative local configuration file can be specified with the --config <file> option.

The specificity is as follows, so that a command line argument will always override the config file settings:

* command line arguments
* local config
* global config

A config file can take any of the command line arguments as JSON key values, for example:

{

"verbose": true,

"ignore": ["\*.test.js", "\*\*/fixtures/\*\*"],

"execMap": {

"rb": "ruby",

"pde": "processing --sketch={{pwd}} --run"

}

}

The above nodemon.json file might be my global config so that I have support for ruby files and processing files, and I can run nodemon demo.pde and nodemon will automatically know how to run the script even though out of the box support for processing scripts.

A further example of options can be seen in [sample-nodemon.md](https://github.com/remy/nodemon/blob/master/doc/sample-nodemon.md)

### package.json

If you want to keep all your package configurations in one place, nodemon supports using package.json for configuration. Specify the config in the same format as you would for a config file but under nodemonConfig in the package.json file, for example, take the following package.json:

{

"name": "nodemon",

"homepage": "http://nodemon.io",

"...": "... other standard package.json values",

"nodemonConfig": {

"ignore": ["\*\*/test/\*\*", "\*\*/docs/\*\*"],

"delay": 2500

}

}

Note that if you specify a --config file or provide a local nodemon.json any package.json config is ignored.

*This section needs better documentation, but for now you can also see nodemon --help config (*[*also here*](https://github.com/remy/nodemon/blob/master/doc/cli/config.txt)*)*.

## Using nodemon as a module

Please see [doc/requireable.md](http://doc/requireable.md)

## Using nodemon as child process

Please see [doc/events.md](http://doc/events.md#Using_nodemon_as_child_process)

## Running non-node scripts

nodemon can also be used to execute and monitor other programs. nodemon will read the file extension of the script being run and monitor that extension instead of .js if there's no nodemon.json:

nodemon --exec "python -v" ./app.py

Now nodemon will run app.py with python in verbose mode (note that if you're not passing args to the exec program, you don't need the quotes), and look for new or modified files with the .py extension.

### Default executables

Using the nodemon.json config file, you can define your own default executables using the execMap property. This is particularly useful if you're working with a language that isn't supported by default by nodemon.

To add support for nodemon to know about the .pl extension (for Perl), the nodemon.json file would add:

{

"execMap": {

"pl": "perl"

}

}

Now running the following, nodemon will know to use perl as the executable:

nodemon script.pl

It's generally recommended to use the global nodemon.json to add your own execMap options. However, if there's a common default that's missing, this can be merged in to the project so that nodemon supports it by default, by changing [default.js](https://github.com/remy/nodemon/blob/master/lib/config/defaults.js) and sending a pull request.

## Monitoring multiple directories

By default nodemon monitors the current working directory. If you want to take control of that option, use the --watch option to add specific paths:

nodemon --watch app --watch libs app/server.js

Now nodemon will only restart if there are changes in the ./app or ./libs directory. By default nodemon will traverse sub-directories, so there's no need in explicitly including sub-directories.

Nodemon also supports unix globbing, e.g --watch './lib/\*'. The globbing pattern must be quoted. For advanced globbing, [see picomatch documentation](https://github.com/micromatch/picomatch#advanced-globbing), the library that nodemon uses through chokidar (which in turn uses it through anymatch).

## Specifying extension watch list

By default, nodemon looks for files with the .js, .mjs, .coffee, .litcoffee, and .json extensions. If you use the --exec option and monitor app.py nodemon will monitor files with the extension of .py. However, you can specify your own list with the -e (or --ext) switch like so:

nodemon -e js,pug

Now nodemon will restart on any changes to files in the directory (or subdirectories) with the extensions .js, .pug.

## Ignoring files

By default, nodemon will only restart when a .js JavaScript file changes. In some cases you will want to ignore some specific files, directories or file patterns, to prevent nodemon from prematurely restarting your application.

This can be done via the command line:

nodemon --ignore lib/ --ignore tests/

Or specific files can be ignored:

nodemon --ignore lib/app.js

Patterns can also be ignored (but be sure to quote the arguments):

nodemon --ignore 'lib/\*.js'

**Important** the ignore rules are patterns matched to the full absolute path, and this determines how many files are monitored. If using a wild card glob pattern, it needs to be used as \*\* or omitted entirely. For example, nodemon --ignore '\*\*/test/\*\*' will work, whereas --ignore '\*/test/\*' will not.

Note that by default, nodemon will ignore the .git, node\_modules, bower\_components, .nyc\_output, coverage and .sass-cache directories and *add* your ignored patterns to the list. If you want to indeed watch a directory like node\_modules, you need to [override the underlying default ignore rules](https://github.com/remy/nodemon/blob/master/faq.md#overriding-the-underlying-default-ignore-rules).

## Application isn't restarting

In some networked environments (such as a container running nodemon reading across a mounted drive), you will need to use the legacyWatch: true which enables Chokidar's polling.

Via the CLI, use either --legacy-watch or -L for short:

nodemon -L

Though this should be a last resort as it will poll every file it can find.

## Delaying restarting

In some situations, you may want to wait until a number of files have changed. The timeout before checking for new file changes is 1 second. If you're uploading a number of files and it's taking some number of seconds, this could cause your app to restart multiple times unnecessarily.

To add an extra throttle, or delay restarting, use the --delay command:

nodemon --delay 10 server.js

For more precision, milliseconds can be specified. Either as a float:

nodemon --delay 2.5 server.js

Or using the time specifier (ms):

nodemon --delay 2500ms server.js

The delay figure is number of seconds (or milliseconds, if specified) to delay before restarting. So nodemon will only restart your app the given number of seconds after the *last* file change.

If you are setting this value in nodemon.json, the value will always be interpreted in milliseconds. E.g., the following are equivalent:

nodemon --delay 2.5

{

"delay": 2500

}

## Gracefully reloading down your script

It is possible to have nodemon send any signal that you specify to your application.

nodemon --signal SIGHUP server.js

Your application can handle the signal as follows.

process.once("SIGHUP", function () {

reloadSomeConfiguration();

})

Please note that nodemon will send this signal to every process in the process tree.

If you are using cluster, then each workers (as well as the master) will receive the signal. If you wish to terminate all workers on receiving a SIGHUP, a common pattern is to catch the SIGHUP in the master, and forward SIGTERM to all workers, while ensuring that all workers ignore SIGHUP.

if (cluster.isMaster) {

process.on("SIGHUP", function () {

for (const worker of Object.values(cluster.workers)) {

worker.process.kill("SIGTERM");

}

});

} else {

process.on("SIGHUP", function() {})

}

## Controlling shutdown of your script

nodemon sends a kill signal to your application when it sees a file update. If you need to clean up on shutdown inside your script you can capture the kill signal and handle it yourself.

The following example will listen once for the SIGUSR2 signal (used by nodemon to restart), run the clean up process and then kill itself for nodemon to continue control:

// important to use `on` and not `once` as nodemon can re-send the kill signal

process.on('SIGUSR2', function () {

gracefulShutdown(function () {

process.kill(process.pid, 'SIGTERM');

});

});

Note that the process.kill is *only* called once your shutdown jobs are complete. Hat tip to [Benjie Gillam](http://www.benjiegillam.com/2011/08/node-js-clean-restart-and-faster-development-with-nodemon/) for writing this technique up.

## Triggering events when nodemon state changes

If you want growl like notifications when nodemon restarts or to trigger an action when an event happens, then you can either require nodemon or add event actions to your nodemon.json file.

For example, to trigger a notification on a Mac when nodemon restarts, nodemon.json looks like this:

{

"events": {

"restart": "osascript -e 'display notification \"app restarted\" with title \"nodemon\"'"

}

}

A full list of available events is listed on the [event states wiki](https://github.com/remy/nodemon/wiki/Events#states). Note that you can bind to both states and messages.

## Pipe output to somewhere else

nodemon({

script: ...,

stdout: false // important: this tells nodemon not to output to console

}).on('readable', function() { // the `readable` event indicates that data is ready to pick up

this.stdout.pipe(fs.createWriteStream('output.txt'));

this.stderr.pipe(fs.createWriteStream('err.txt'));

});

## Using nodemon in your gulp workflow

Check out the [gulp-nodemon](https://github.com/JacksonGariety/gulp-nodemon) plugin to integrate nodemon with the rest of your project's gulp workflow.

## Using nodemon in your Grunt workflow

Check out the [grunt-nodemon](https://github.com/ChrisWren/grunt-nodemon) plugin to integrate nodemon with the rest of your project's grunt workflow.

## Pronunciation

nodemon, is it pronounced: node-mon, no-demon or node-e-mon (like pokémon)?

Well...I've been asked this many times before. I like that I've been asked this before. There's been bets as to which one it actually is.

The answer is simple, but possibly frustrating. I'm not saying (how I pronounce it). It's up to you to call it as you like. All answers are correct :)

## Design principles

* Fewer flags is better
* Works across all platforms
* Fewer features
* Let individuals build on top of nodemon
* Offer all CLI functionality as an API
* Contributions must have and pass tests

Nodemon is not perfect, and CLI arguments has sprawled beyond where I'm completely happy, but perhaps it can be reduced a little one day.

## FAQ

See the [FAQ](https://github.com/remy/nodemon/blob/master/faq.md) and please add your own questions if you think they would help others.

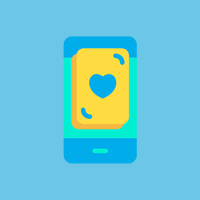
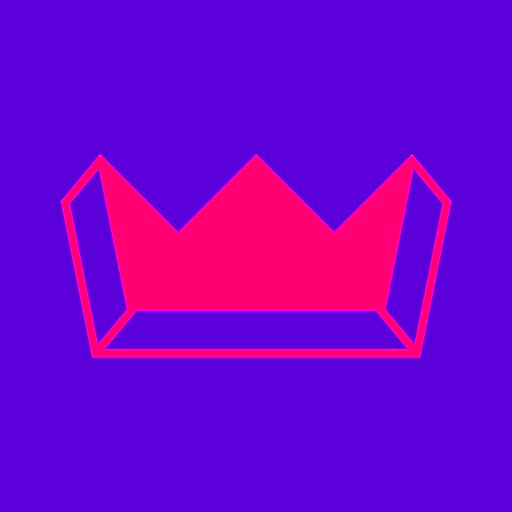
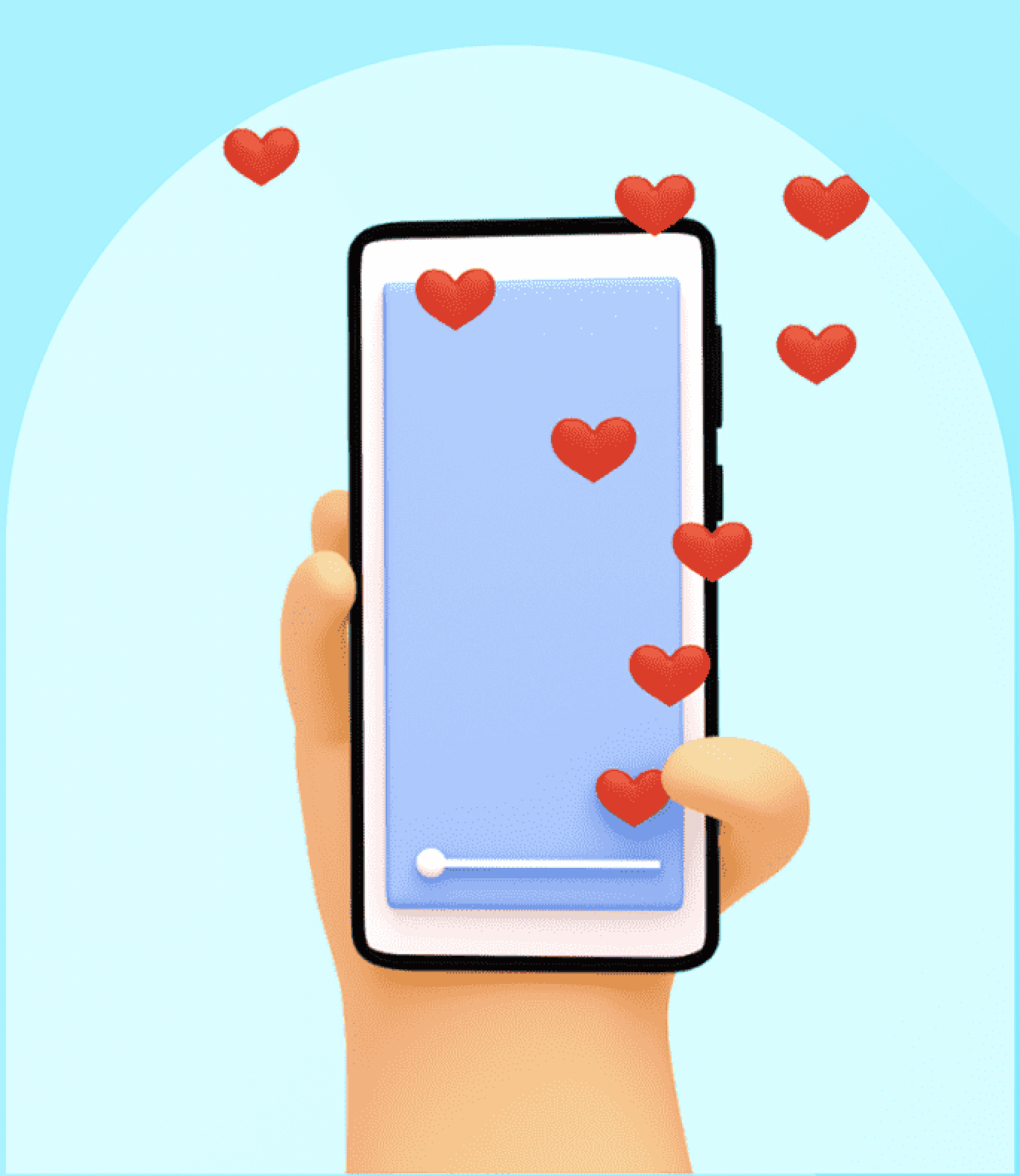
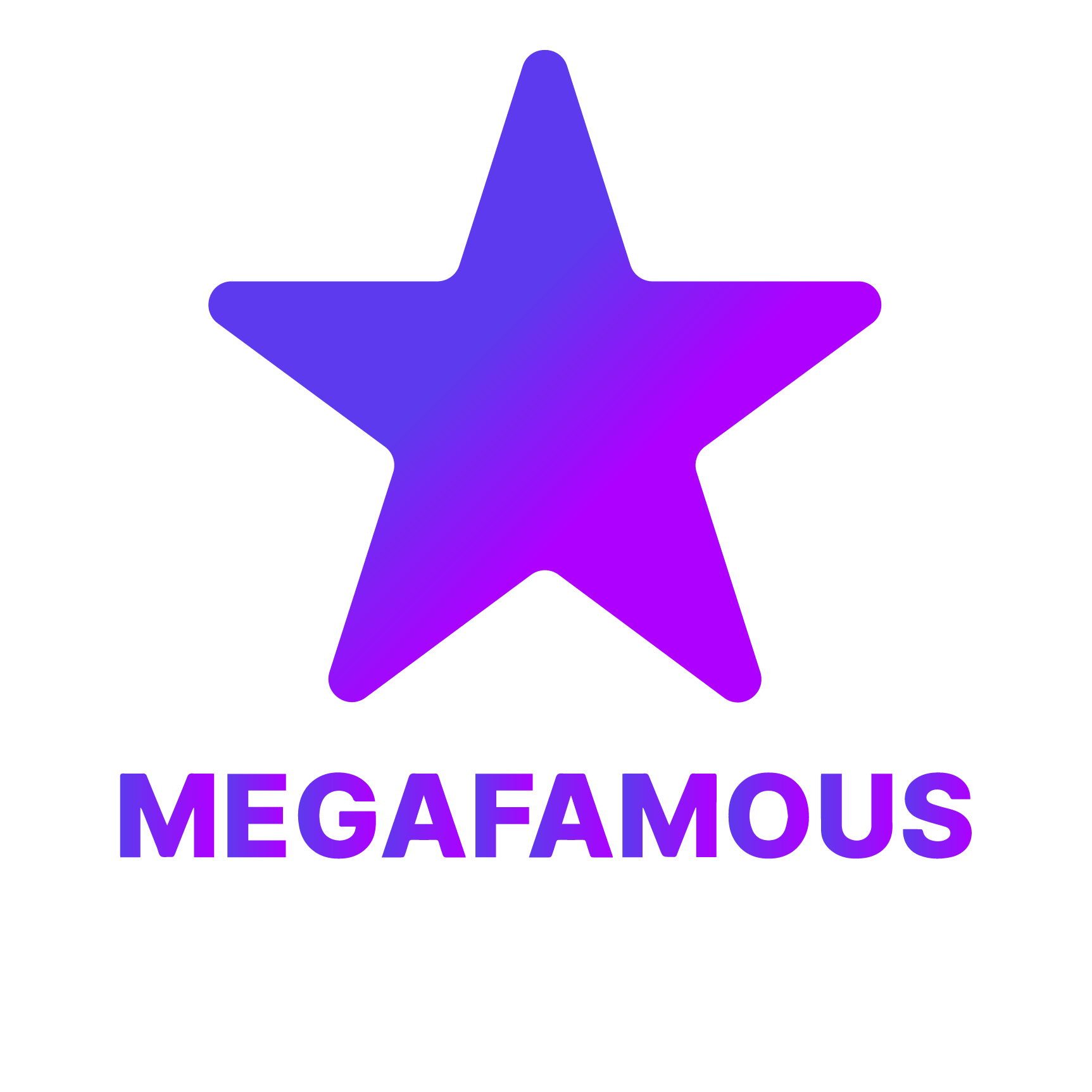
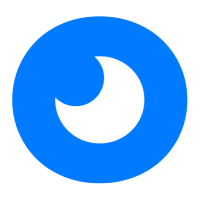
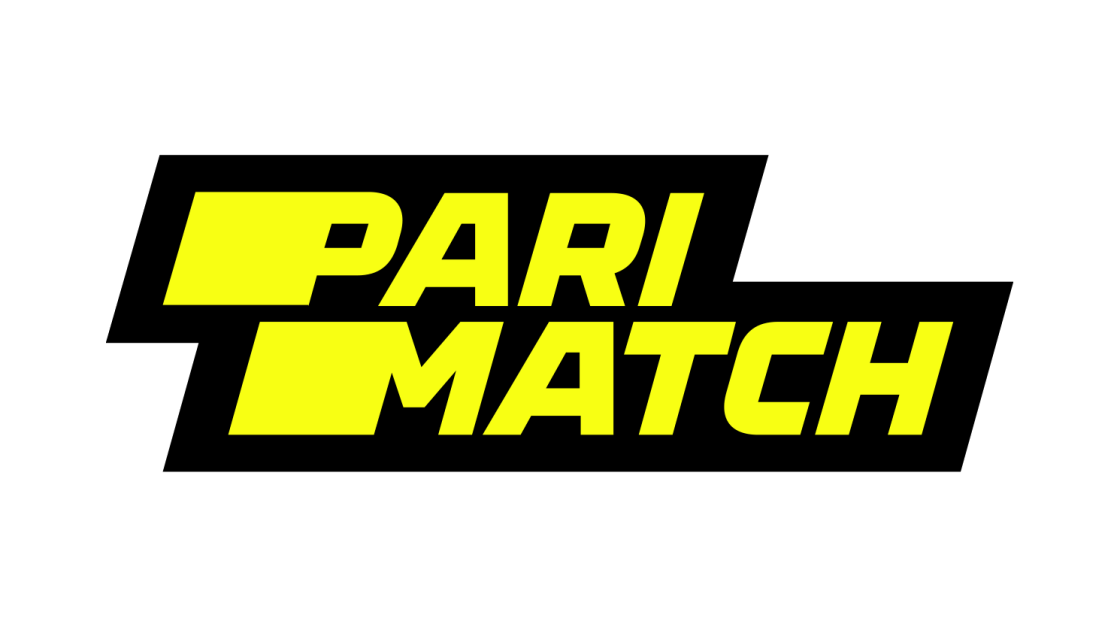
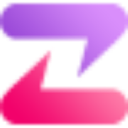
## Backers

Thank you to all [our backers](https://opencollective.com/nodemon#backer)! 🙏

[nodemon backers](https://opencollective.com/nodemon#backers)

## Sponsors

Support this project by becoming a sponsor. Your logo will show up here with a link to your website. [Sponsor this project today ❤️](https://opencollective.com/nodemon#sponsor)

 [](https://najlepsibukmacherzy.pl/ranking-legalnych-bukmacherow/) [](https://pl.polskiekasynohex.org/) [](https://www.casinoonlineaams.com) [](https://casino-wise.com/) [](https://www.thecasinodb.com) [](https://www.casinoutansvenskalicensen.se/) [](https://hellsbet.com/en-au/) [](https://inkedin.com) [](https://www.casinostranieri.net/) [](https://goread.io/buy-instagram-followers) [](https://www.correctcasinos.com/australian-online-casinos/) [](https://casinoburst.com/casino-utan-licens/) [](https://www.uudetkasinot.com) [](https://www.noneedtostudy.com/take-my-online-class/) [](https://www.slotmachineweb.com/) [](https://www.ghotala.com/) [](https://www2.italianonlinecasino.net/) [](https://nongamstopcasinos.net/gb/) [](https://www.scommesse777.com/) [](https://twicsy.com/buy-instagram-likes) [At Casinoaustraliaonline.com, we review, compare and list all the best gambling sites for Aussies.
](https://www.casinoaustraliaonline.com/under-1-hour-withdrawal-casinos/) [Casinon utan svensk licens erbjuder generösa bonusar och kampanjer. Besök coolspins.net för att utforska säkra och pålitliga alternativ.](https://coolspins.net/) [](https://bestuscasinos.org) [](https://www.tightpoker.com/) [](https://poprey.com/) [](https://www.nettcasino.com/) [](https://onlinecasinosspelen.com) [](https://nieuwe-casinos.net) [](https://casinozonderregistratie.net/) [](https://www.vedonlyontibonukset.com/pitkavetovihjeet) [](https://famoid.com/) [](https://www.outlookindia.com/outlook-spotlight/slots-not-on-gamstop-new-non-gamstop-casinos-uk-news-284058) [](https://slotoking.ua/games/all-slots/) [](https://www.bairesdev.com/sponsoring-open-source-projects/) [](https://twicsy.com/buy-instagram-followers) [](https://www.socialwick.com/instagram/followers) [](https://www.onlineunitedstatescasinos.com/) [](https://aviators.com.br) [Online iGaming platform with reliable and trusted reviews.](https://onlinecasinohex.ph/) [](https://online-casinosaustralia.com/) [](https://views4you.com/buy-youtube-subscribers/) [](https://buycheapestfollowers.com/buy-telegram-channel-members) [](https://casinolandia.com) [](https://casino-utan-svensk-licens.com/) [](https://www.slotozilla.com/au/free-spins) [](https://aussiecasinoreviewer.com/) [](https://megafamous.com/buy-automatic-instagram-likes) [](https://popularitybazaar.com/instagram-likes/) [](https://uk.nonstopcasino.org/non-gamstop-casinos/) [](https://philippinescasinos.ph/gcash/) [](https://www.outlookindia.com/outlook-spotlight/casinos-not-on-gamstop-uk-news-302214) [](https://www.nongamstopbets.com/casinos-not-on-gamstop/) [](https://www.stormlikes.net/buy-instagram-followers) [](https://www.upgrow.com/) [Analysis of payment methods for use in the iGaming](https://payidpokies.net/) [30 Best Casinos Not on Gamstop in 2024](https://finance.yahoo.com/news/30-best-casinos-not-gamstop-091943696.html) [No deposit casino promo Codes 2024 - The best online Casinos websites. No deposit bonus codes, Free Spins and Promo Codes. Stake, Roobet, Jackpotcity and more.](https://www.ownedcore.com/casino) [](https://www.fruityking.co.nz) [](https://www.nodepositcasinobonus.org/) [Fortune Tiger](https://fortune-tiger-br.com/) [](https://parimattchbr.com/) [](https://expressfollowers.com/buy-instagram-reel-views/) [](https://sidesmedia.com) [](https://buitenlandseonlinecasinos.org/) [](https://bulkoid.com/buy-twitter-followers) [](https://viralyft.com/buy-youtube-views) [](https://graming.com/tiktok-views/) [](https://insfollowpro.com) [](https://www.casinoonlineca.ca/)

Please note that links to the sponsors above are not direct endorsements nor affiliated with any of contributors of the nodemon project.

# License

MIT <http://rem.mit-license.org>