nodemon Restart on file change

CONCEPTS

nodemon is a restart loop: it runs your command, waits for a watched file to change, kills the process and starts it again.

Start run command
↓ app runs
Watch files by ext/path
↓ file saved
Debounce --delay
↓ signal
Restart same command

Without --exec nodemon runs node (or the mapped runner for the script's extension, e.g. python for .py).

Defaults: watches the current directory for js, mjs, json and coffee; ignores .git and node_modules.

RUNNING

`nodemon <script.js>`
Run script, restart on changes

`nodemon`
Run main or scripts.start of package.json

`nodemon <script.js> <arg1> <arg2>`
Pass arguments to the script

`nodemon <script.js> -- --port 3000`
Stop nodemon parsing; pass flags on

`nodemon --inspect <script.js>`
Restart with the Node debugger

`nodemon --exec "<cmd>"`
Run any command instead of node

`nodemon -x python <app.py>`
Run a Python script

`nodemon -x "npx ts-node" <app.ts>`
Run TypeScript through ts-node

`nodemon -C <script.js>`
Start only after the first change

RUNNING (cont.)

`nodemon --cwd <dir> <script.js>`
Run from another directory

WATCHING

`nodemon -w <dir> <script.js>`
Watch a directory or file

`nodemon -w <a> -w <b> <script.js>`
Watch several paths

`nodemon -e js,json,html <script.js>`
Watch these extensions

`nodemon -i <path>`
Ignore a file or directory

`nodemon -i "*.test.js" <script.js>`
Ignore a glob (quote it)

`nodemon -i "docs/**" <script.js>`
Ignore a whole tree

`nodemon -d <seconds> <script.js>`
Debounce restarts

`nodemon -L <script.js>`
Poll for changes (Docker, network drives)

Quote globs, or the shell expands them before nodemon sees them.

CONTROL

`rs`
Restart manually (type, then Enter)

`Ctrl+C`
Quit nodemon and the app

`nodemon -V <script.js>`
Show which file caused a restart

`nodemon --dump`
Print the effective configuration

`nodemon --help options`
List all options

A clean exit prints 'clean exit - waiting for changes'; a crash waits for the next change too.

CONFIGURATION

`nodemon.json`
Per-project config next to the code

`nodemonConfig in package.json`
Same keys, inside package.json

`nodemon --config <file>`
Use another config file

`$HOME/nodemon.json`
Global defaults for all projects

Common config keys

Key	Value	Matches
`watch`	array of paths	`-w`
`ext`	"js,json"	`-e`
`ignore`	array of globs	`-i`
`exec`	command string	`-x`
`delay`	milliseconds	`-d`
`verbose`	true / false	`-V`
`legacyWatch`	true / false	`-L`
`env`	object of variables	`(config only)`
`execMap`	extension to runner	`(config only)`
`events`	event to shell command	`(config only)`

Precedence: CLI arguments beat the local config, which beats the global one.

RECIPES

Project nodemon.json

```
{
  "watch": ["src"],
  "ext": "ts,json",
  "ignore": ["src/**/*.spec.ts"],
  "exec": "ts-node src/index.ts",
  "delay": 500,
  "env": { "NODE_ENV":
"development" }
}
```

Run a different language

```
nodemon --exec python3 app.py

# or map extensions in
nodemon.json
{ "execMap": { "py": "python3",
"rb": "ruby" } }
```

Run a build or test on change

```
nodemon -w src -e js --exec "npm
test"
nodemon -w src -e ts --exec "npm
run build"
```