Learn pm2

❤️Docs: https://pm2.keymetrics.io/docs/usage/quick-start
- Todo: Read the docs from above link
- ❤️LEARN: pm2 has a built in load balancer as well ❤️ - (check in above link) and explore it.

TODO:
- **TODO_READ**: Restart strategies
- Setting up PM2 CI deployments with Github Actions: Click here
- Pm2 github Actions App: Click here

Why the FUCK `pm2 stop` or `pm2 stop all` removes the watching option of (every) app (checked via `pm2 l` command)?

ChatGPT
tldr; It does, fuck it!
Simply use your alias command `pm2.start` which is aliased to `pm2 start 1 2 4 6 7 --watch` which starts all apps in watch mode. 🎉.

❤️Checking CPU & Memory consumption of pm2 apps

🔴TODO - CHECK WHICH APP USES HIGH CPU.

🔴TODO - LIMIT THE APP WHICH USES MORE RAM THAN NEEDED BY --max-memory-restart FLAG.

Oct 18, 2025
Source: ChatGPT

```
pm2 list
pm2 monit # for live cpu and memory

# Restart app if it consumes more than x amount of ram (memory):
pm2 start app.js --name "my-app" --max-memory-restart 200M

# If the app is already running:
pm2 restart my-app --max-memory-restart 200M
```

```
# or update it permanently:
pm2 set pm2:my-app:max_memory_restart 200M

# To make it persist across reboots:
pm2 save
```

Get in JSON:
*Note: The cpu is in percentage i.e, 1.2 % and memory in bytes i.e., 88346624 / (1024*1024) = 84mb.*

```
pm2 jlist
# Shows complete information about each application here...

# To get only cpu and memory:
pm2 jlist | jq '.[] | {name: .name, cpu: .monit.cpu, memory:
.monit.memory}'
Output:
{
  "name": "staging-qr-backend",
  "cpu": 1.2, // 1.2 %
  "memory": 88346624 // 84mb    [memory / (1024*1024)]
}
{
  "name": "prod-qr-backend",
  "cpu": 1.6,
  "memory": 88346624
}
{
  "name": "prod-qr-frontend",
  "cpu": 1.7,
  "memory": 90853376
}
```

❤️In case you restarted your system and you haven't had run command - `pm2 startup` you can still restore all processes via

TODO- MOVE-THIS-IN-"Restart persistence" heading also, in the end.

[Tested on linode archlinux server]

```
pm2 resurrect
```

❤️❤️Restart persistence

On linode VPS:

```
# On running below command - it printed a command to the console asking to copy-paste and run the command
pm2 startup
# After running that command I got output as:

# You can freeze a process list on reboot via:
pm2 save
# You can remove init script via:
pm2 unstartup systemd
```

🚀Pm2's Deployment System & ❤️Configuration File

Docs:

- Deployment System: Click here
- Configuration File: Click here
- Config file Syntax: Click here

❤️❤️🚀✅**Check file ecosystem.config.js file in your qr-backend project.**

- *Why use a config file (^) instead of using inline pm2 command to create apps? (**Check comments in above file**).*

Notes from Sahil:

- ⭐This deployment system is useful when you want to do a *Multi host deployment* with pm2's way.

Note: Make sure the application configuration file in the local folder is named either `ecosystem.config.js` or `pm2.config.js`, so you don't need to type the configuration filename for each command. (Check the above docs link to know example config file contents).

```
# Go to application folder
cd my-app
# Generate ecosystem config file and update it as needed
pm2 init # or use pm2 ecosystem
# Run remote setup commands
pm2 deploy production setup
# Deploy app
pm2 deploy production
# Note: if git reports an error that there are local changes but still
```

```
wants to push what is on the remote GIT, you can use the --force option to
force deployment.
pm2 deploy production --force
# Rollback to previous deployment
pm2 deploy production revert 1
```

✅Node.js, the difference between development and production

Awesome: Click here ❤️Nodejs Docs says that using values other than “production” for NODE_ENV is an antipattern.

- https://12factor.net (sound like a good read)

❤️ 0-seconds downtime reload using “graceful start”

❤️Docs:

- **TODO** Find the heading “Graceful start” on this page.
- **TODO** Cluster Mode: Click here
- **TODO** Load-Balancing (cluster mode): Click here

Sahil’s Notes - You need to use:

1. send “ready” process event (check docs link below)
2. Use cluster mode
3. Use `pm2 reload appName` (instead of `pm2 restart appName`)
    a. Check this Stackoverflow answer to know `reload` vs. `restart` in pm2.

❤️HORIZONTAL SCALING

HORIZONTAL:

- I can use the “instance” feature of pm2 to horizontal scale + increase the CPU & RAM of the VPS.
- I can use cloudflare loadbalancing to divert traffic between multiple VPS.

You can define instances (number of replicates for an app) for load balancing (probably)

❤️**TODO**: Check this out in future when have scaling issues.

❤️I have app ids in `pm2 ls` output as 1 and 3, can I update them 0 and 1 to look more sequential? (#id, #irregular, #random)

✅Yes, you can simply run the `pm2 update` command. It restarts the PM2 daemon and reloads all processes with the latest configuration and environment. It's useful after changes to ecosystem files or environment variables.
You can verify if apps are restarted after running **`pm2 update`** simply by looking at their "uptime" value if you run `pm2 ls` command and it'll show you a few seconds only since all apps just restarted. (source)

😎Irregular ids issue of apps in pm2:

```
pm2 start app1.js   # ID 0
pm2 start app2.js   # ID 1
pm2 delete 1
pm2 start app3.js   # ID 2 (not 1 again)
```

Why PM2 IDs Skip Numbers:

1. Persistent Internal ID Counter: PM2 maintains an internal counter for assigning process IDs. Even if you delete a process, the counter does not reset or reuse that ID.
2. Design Choice for Stability: This design prevents potential issues that could arise from ID reuse, such as conflicts in process monitoring, logging, or automation scripts.
3. Metadata Tracking: PM2 keeps metadata in ~/.pm2 or similar storage, and recycling IDs could complicate tracking logs and historical data.

❤️Can I restart apps when files are changed in app directories as well as when `git pull` is done?

Yes, pm2 has a "watching" option, you can enable it!

Can I set pm2 to fetch code from git automatically?

TODO: MOVE THIS CONTENT TO where you have “pm2 deploy” code instructions -- for easy comparability.

ChatGPT, another way to do this is to add a cronjob to pull every minute.
Pm2 deploy seems the way to deploy on remote servers — check notes above for this.

What is the use of pm2? Give me an example usage where I need pm2? (ChatGPT

PM2 is a process manager for Node.js applications that helps you manage and monitor applications in production. It ensures that your application runs continuously, restarts on crashes, and can be easily managed with logs, monitoring, and auto-start on system boot.

Use Cases for PM2:

1. **Keeping Applications Running:** If your Node.js app crashes or the server restarts, PM2 will automatically restart it.
2. **Load Balancing:** PM2 can run multiple instances of an application to distribute traffic.
3. **Process Monitoring:** It provides real-time monitoring of CPU, memory usage, and logs.
4. **Log Management:** PM2 collects logs, making debugging easier.
5. **Startup Scripts:** It can automatically start your app when the system boots.

What does pm2 stand for?
PM2 stands for **"Process Manager 2"**. It is the second version of a process manager designed specifically for Node.js applications, though it can also manage other scripts like Python, Bash, and more.

The name **PM2** reflects its core functionality: managing and monitoring processes to ensure high availability, automatic restarts, and efficient resource usage.

Pm2 Usage

```
# install globally (system-wide)
npm i -g pm2

# initialize pm2
pm2 list
# or use alias
pm2 ls
```

or use alias ✅
```
pm2 l
```

start application
```
pm2 start server-a.js
```

❤️❤️ Set memory threshold for app reload
```
--max-memory-restart <200MB>
```

start application with specified name (Otherwise default name server-a [name of program file is used as application name])
```
pm2 start server-a.js --name my-server
```

 # ❤️Create a pm2 app with arguments to command:
Learn: Using nodemon, we pass `--exitcrash` to nodemon so that nodemon exits and pm2 restarts the server else our server remains crashed):
```
pm2 start nodemon --name my-app -- --exitcrash server-a.js
```
NOTE: If you already have a nodemon app which you started without --exitcrash option then you can simply update the args of that app by: [TESTED]
```
pm2 restart <app_name_or_id> -- --exitcrash server-a.js
```
For command: “npm start”
```
pm2 start npm --name my-app -- start
```
For command “npm run dev”
```
pm2 start npm --name new-ai-sdk -- run dev
```

 # ❤️Giving inline environment variables
```
NODE_ENV="myenv" pm2 start ./app.js
```
Note: If you use below way then the pm2 fails with error:
```
pm2 start NODE_ENV="myenv" ./app.js
```
[PM2][ERROR] Script not found: /Users/apple/Documents/test/test-pm2/NODE_ENV=myenv

 # ❤️❤️ Managing different set of environment variables using ecosystem file: (check docs here)

Start an app
```
pm2 start app.js
```
Or start any other application easily:
```
pm2 start bashscript.sh
pm2 start python-app.py --watch
pm2 start binary-file -- --port 1520
```

❤️ Prefix logs with time
`--time`

❤️Do not auto restart app
`--no-autorestart`

❤️Specify cron for forced restart
`--cron <cron_pattern>`

start application with custom name and specify log file which has both stdout and stderr via **`--log`** option
`pm2 start "npx nps" --name qr-backend-dev --log ~/.pm2/logs/qr-backend-dev-out-error.log`
Note: By default pm2 does not generate a combined log output file ☹️so we need to use `--log` option.
Note: In addition to file **`qr-backend-dev-out-err.log`**, there are two other files - `qr-backend-dev-out.log` (STDOUT) and `qr-backend-dev-err.log` (STDERR) in `~/.pm2/logs/` directory.
Note: You can specify the stdout and stderr files path via `--output` and `--error` respectively in addition to the `--log` option.

start with `--watch` option to restart server whenever any file changes *(if you don't pass any argument to --watch flag then current directory is watched)*
`pm2 start "npx nps" --name qr-backend-dev --log ~/.pm2/logs/qr-backend-dev-out-error.log --watch`
Note: If you are specifying `--watch` option then you must not put log files in the current directory otherwise pm2 goes into an infinite restart loop for the application. So either use the default logs directory for log files or you can use the parent directory of the application to your log files.
Note: You can use `--ignore-watch` to ignore any directory e..g, `--ignore-watch "node_modules"`

In above command if you specify log type as json via option `--log-type json` then you'll see logs format (in all three logs file) as:

```
{
  "message": "MONGO DB CONNECTED: 127.0.0.1\n",
  "timestamp": "2025-02-16T16:33:03.495Z",
  "type": "out",
  "process_id": 0,
  "app_name": "qr-backend-dev"
}
{
  "message": "🔴 ERROR MIDDLEWARE: 16/2/2025, 10:04:15 pm \n Error: Error: Coffee Not Found ☕🚫\n    at /Users/apple/Documents/test/qr-solution/qr-solution-backend/src/app.ts:26:35\n    at Layer.handle [as handle_request] (/Users/ap…",
  "timestamp": "2025-02-16T16:34:15.946Z",
  "type": "err",
  "process_id": 0,
  "app_name": "qr-backend-dev"
}
```

*Note: This is helpful if you like the above format of logs and want to see **timestamp** for each command as you see in the above command.*
From ChatGPT: ***When to Use JSON Logs?***
- ✅ When integrating with log management systems.
- ✅ When you need structured logs instead of plain text.
- ✅ When parsing logs programmatically for debugging or analytics.
- Also: The logs look good in terminal when you see them via — `cat /Users/apple/.pm2/logs/qr-backend-dev-out-err.log | jq`

Note to Sahil: I do not want to use json format logs because it restricts my readability to see logs and errors on any production server. It might be useful if i want to build some ui to show logs there instead of seeing logs on a hosted ec2 instance.

🌟 view all applications
```
pm2 ls
```

🚀 restart application
```
pm2 restart all # to restart all apps
pm2 restart 1 # 1 is the id of your app shown in `pm2 ls` command
# or you can use app name too:
pm2 restart my-server
# or you can restart multiple apps by providing multiple ids/names of apps
# to restart the apps with id 1 and 3.
pm2 restart 1 3
```

🛑 stop application
pm2 stop all # to stop all apps
```
pm2 stop 1 # 1 is the id of your app shown in `pm2 ls` command
# or you can use app name too:
pm2 stop my-server
```

✅ start application
pm2 start all # to start all apps
```
pm2 start 1 # 1 is the id of your app shown in `pm2 ls` command
# or you can use app name too:
pm2 start my-server
```

❌ delete application
pm2 delete all # To delete all apps
```
pm2 delete 1 # 1 is the id of your app shown in `pm2 ls` command
# or you can use app name too:
pm2 delete my-server
```

view logs of all application
```
pm2 logs
```
Output logs files:
```
~/.pm2/logs/qr-backend-dev-out.log
~/.pm2/logs/qr-backend-dev-error.log
```
(optional if you specify via `--log` option): `~/.pm2/logs/qr-backend-dev-error.log`

view logs of particular application

```
pm2 logs 1 # 1 is the id of your app shown in `pm2 ls` command
# or you can use app name too:
pm2 logs my-server
pm2 logs my-server --lines 10000
```

view start, stop, restart logs of application

```
pm2 logs --timestamp qr-backend-dev
```

 # 🚀🚀🚀 Save and resurrect list of apps

`pm2 save` # saves current list of apps to **~/.pm2/dump.pm2**

`pm2 delete 1` (or you may delete all apps with `pm2 delete all`)

Restores all apps along with their start/stop status from list of apps defined in **~/.pm2/dump.pm2** [TESTED]

```
pm2 resurrect
```

—---------- help commands

```
pm2 start -h
pm2 logs -h
pm2 stop -h
```

—---

✅🚀❤️ show info about app

```
pm2 show qr-backend-dev
```

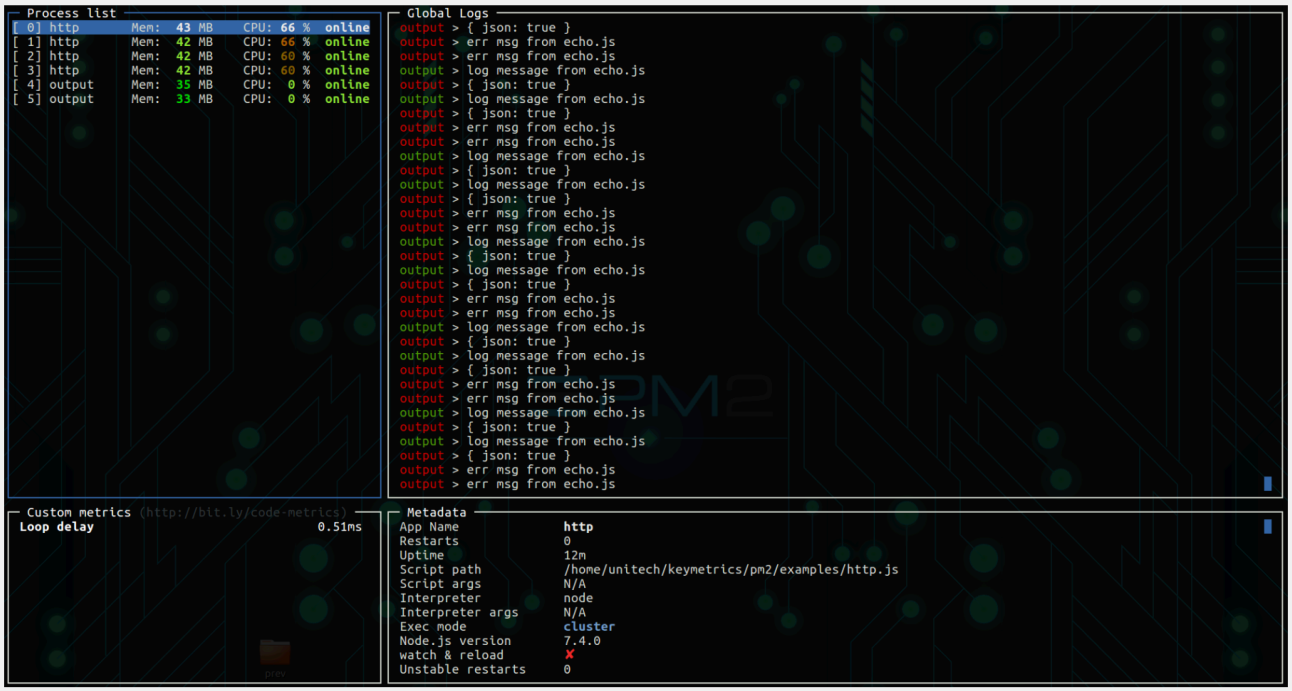
(Note: The file `/Users/apple/.pm2/pids/qr-backend-dev-0.pid` mentioned for `pid path` has content as process id of this process.

```
apple@apples-MacBook-Pro qr-solution % pm2 show qr-backend-dev
 Describing process with id 0 - name qr-backend-dev

 status                 online
 name                   qr-backend-dev
 namespace              default
 version                N/A
 restarts               3
 uptime                 47s
 script path            /bin/bash
 script args            -c npx nps
 error log path         /Users/apple/.pm2/logs/qr-backend-dev-error.log
 out log path           /Users/apple/.pm2/logs/qr-backend-dev-out.log
 pid path               /Users/apple/.pm2/pids/qr-backend-dev-0.pid
 interpreter            none
 interpreter args       N/A
 script id              0
 exec cwd               /Users/apple/Documents/test/qr-solution/qr-solution-backend
 exec mode              fork_mode
 node.js version        N/A
 node env               N/A
 watch & reload         ✔
 unstable restarts      0
 created at             2025-02-16T14:58:28.464Z

 Divergent env variables from local env

 PATH      /Users/apple/.nvm/versions/node/v22.13.0/bi
 PWD       /Users/apple/Documents/test/qr-solution/qr-
 MANPATH   /Users/apple/.nvm/versions/node/v22.13.0/sh

  Add your own code metrics: http://bit.ly/code-metrics
  Use `pm2 logs qr-backend-dev [--lines 1000]` to display logs
  Use `pm2 env 0` to display environment variables
  Use `pm2 monit` to monitor CPU and Memory usage qr-backend-dev
```

❤️Docs: https://pm2.keymetrics.io/docs/usage/monitoring/

pm2 monit

Monitoring CPU/Memory



Make pm2 auto-boot at server restart:

enable running pm2 apps start server startup/restart/reboot

```
pm2 startup
```

- The `pm2 startup` command generates and configures a startup script to ensure that PM2 (and the processes it manages) automatically restart when your server reboots.

Save current list of current pm2 apps so that pm2 apps start on server startup/restart/reboot

```
pm2 save
```

When do I need to run `pm2 save` command:

1. **After Adding a New Application:** For example, if you start a new app with:
   `pm2 start newApp.js --name new-app`
2. **After Removing or Restarting an Application:** Similarly, if you delete or update an application, run `pm2 save` to update the snapshot.

Custom Solution to Manage Processes (Sahil)

If some day I want to move away from pm2 I can make use of something like this - Create folder .ignored in backend in qr solution and make a .keep file in it and make folder runningProcesses folder (with .keep folder in it). And this setup to .gitignore file. Also mentioned in folder runningProcesses/README.md that this folder needs to have files with names like 25644,34644 (i.e., pids of running process of this app) created by the app itself when program starts. And also add a script cleanStaleProcessIds.sh (which cleans processIds which no longer exists). Also add SIGTERM and SIGINT handler which cleans up the process id files from this folder on program exit.