# pm2使用

目录

[一. pm2 安装 1](#_Toc31425)

[二. pm2 使用 2](#_Toc9262)

[1.各类用法： 2](#_Toc23966)

[附录：全部 2](#_Toc20249)

# pm2 安装

直接敲击npm install pm2 -g 即可安装成功命令行界面

ps：如果想要一个可视化UI

$ pm2 web

pm2会启动一个叫做pm2-http-interface的进程提供web服务。你打开浏览器输入http： //127.0.0.1:9615

# pm2 使用

1.各类用法：

我们常用的 应该有

Ps: 0 代表 对应的www文件(express项目)

下面是一个启动时可以使用的json 包含日志输出；

因为我们的项目也有日志输出 所以其实可以不需要这样启动

{

**"name"** : **"subscribe"**,  
 **"script"** : **"./bin/www"**,  
 **"error\_file"** : **"./logs/error.log"**,  
 **"out\_file"** : **"./logs/out.log"**,  
 **"log-date-format"** : **"YYYY-MM-DD HH:mm Z"** }

$ pm2 start 0 --name my-api # 命名进程

$ pm2 list # 显示所有进程状态  
 $ pm2 stop all # 停止所有进程  
 $ pm2 restart all # 重启所有进程  
 $ pm2 reload all # 0秒停机重载进程 (用于 NETWORKED 进程)  
 $ pm2 stop 0 # 停止指定的进程  
 $ pm2 restart 0 # 重启指定的进程  
 $ pm2 delete 0 # 杀死指定的进程  
 $ pm2 delete all # 杀死全部进程

$ pm2 show 0 # 查看指定的进程情况

$ pm2 logs # 查看所有日志

$ pm2 logs [app-name] # 查看指定进程的目录

# 附录：全部

$ pm2 start app.js -i 4 #后台运行pm2，启动4个app.js   
                                # 也可以把'max' 参数传递给 start  
                                # 正确的进程数目依赖于Cpu的核心数目  
$ pm2 start app.js --name my-api # 命名进程  
$ pm2 list # 显示所有进程状态  
$ pm2 monit # 监视所有进程  
$ pm2 logs # 显示所有进程日志  
$ pm2 stop all # 停止所有进程  
$ pm2 restart all # 重启所有进程  
$ pm2 reload all # 0秒停机重载进程 (用于 NETWORKED 进程)  
$ pm2 stop 0 # 停止指定的进程  
$ pm2 restart 0 # 重启指定的进程  
$ pm2 startup # 产生 init 脚本 保持进程活着  
$ pm2 web # 运行健壮的 computer API endpoint ([http://localhost:9615](http://localhost:9615/" \t "https://www.douban.com/note/314200231/_blank))  
$ pm2 delete 0 # 杀死指定的进程  
$ pm2 delete all # 杀死全部进程

运行进程的不同方式：  
$ pm2 start app.js -i max # 根据有效CPU数目启动最大进程数目  
$ pm2 start app.js -i 3 # 启动3个进程  
$ pm2 start app.js -x #用fork模式启动 app.js 而不是使用 cluster  
$ pm2 start app.js -x -- -a 23 # 用fork模式启动 app.js 并且传递参数 (-a 23)  
$ pm2 start app.js --name serverone # 启动一个进程并把它命名为 serverone  
$ pm2 stop serverone # 停止 serverone 进程  
$ pm2 start app.json # 启动进程, 在 app.json里设置选项  
$ pm2 start app.js -i max -- -a 23 #在--之后给 app.js 传递参数  
$ pm2 start app.js -i max -e err.log -o out.log # 启动 并 生成一个配置文件  
你也可以执行用其他语言编写的app ( fork 模式):  
$ pm2 start my-bash-script.sh -x --interpreter bash  
$ pm2 start my-python-script.py -x --interpreter python  
  
0秒停机重载:  
这项功能允许你重新载入代码而不用失去请求连接。  
注意：  
仅能用于web应用  
运行于Node 0.11.x版本  
运行于 cluster 模式（默认模式）  
$ pm2 reload all  
  
CoffeeScript:  
$ pm2 start my\_app.coffee #这就是全部  
  
PM2准备好为产品级服务了吗？  
只需在你的服务器上测试  
$ git clone [https://github.com/Unitech/pm2.git](https://github.com/Unitech/pm2.git" \t "https://www.douban.com/note/314200231/_blank)  
$ cd pm2  
$ npm install # 或者 npm install --dev ，如果devDependencies 没有安装  
$ npm test  
  
pm2 list  
列出由pm2管理的所有进程信息，还会显示一个进程会被启动多少次，因为没处理的异常。

|  |
| --- |
| IMG_256 |
|  |

pm2 monit  
监视每个node进程的CPU和内存的使用情况。

|  |
| --- |
| IMG_257 |