package.json

我们在使用npm install来安装模块或插件的时候，有两种命令把依赖信息写入到package.json文件里面去：

npm i [包名] --save-dev

or

npm i [包名] --save

在package.json文件里面体现出来的区别就是：

使用”npm i [包名] --save-dev”安装的包，会被写入到devDependencies配置中取，而”npm i [包名] --save”安装的包，则会被写入到dependencies配置中取。

# devDependencies

|  |  |
| --- | --- |
| 内容 | 是一个对象，配置模块依赖的模块列表，key是模块名称，value是版本范围。 |
| 作用 | 该模块中所列举的插件属于开发环境的依赖（比如：测试或者文档框架等） |
| 部署来源 | npm i [包名] --save-dev |

# dependencies

|  |  |
| --- | --- |
| 内容 | 是一个对象，配置模块依赖的模块列表，key是模块名称，value是版本范围。 |
| 作用 | 改模块中所列举的插件属于生产环境的依赖（程序正常运行需要加载的依赖） |
| 部署来源 | npm i [包名] --save |

# dependencies vs devDependencies

devDependencies里面的插件（比如各种loader、babel全家桶及各种webpack的插件等）只用于开发环境，不用于生产环境，因此不需要打包；而dependencies里面的插件是需要发布到生产环境的，是要打包的（比如vue、vue-router、vuex、axios等）。

dependencies里面的插件是保证应用能够正常运行的依赖包。不仅开发环境能够使用，生产环境也能使用；devDependencies里面的插件是开发应用时能依赖的工具包。只有本地开发环境能用，且其不会被打包入生产环境。有各种工具，loader包含在内。

# 依赖写入package.json的作用

* 如果拿到别人的项目，需要安装package.json中devDependencies和dependencies两个模块下所列举的依赖，可以通过执行以下命令实现：

npm install

* 如果拿到别人的项目，只需要安装package.json中dependencies模块下所列举的依赖，可以通过执行以下命令实现：

npm install packagename

* 如果拿到别人的项目，只需要安装package.json中devDependencies模块下所列举的依赖，可以通过执行以下命令实现：

npm install packagename -dev

# npm uninstall 删除依赖

* 删除依赖，但不会删除package.json中的配置（即通过npm install依然可以安装该依赖），可以通过执行以下命令实现：

npm uninstall “依赖名称”

* 删除依赖，同时删除package.json中devDependencies的配置，可以通过执行以下命令实现：

npm uninstall “依赖名称” --save-dev

* 删除依赖，同时删除package.json中dependencies的配置，可以通过执行以下命令实现：

npm uninstall “依赖名称” --save

# 提示

编程中如果自己需要某一依赖，在安装时一定要将依赖配置到package.json中，而非简单地使用npm install “依赖名称”，因为这样虽然可以正常安装依赖，但不会配置到package.json文件中，那么，当其他协作人员的项目不能正常运行后不能通过npm install正确安装依赖，别人也并不知道你安装了什么依赖，这样就会大大影响协作编程的效率。