**NPM Command**

NPM是随同NodeJS一起安装的包管理工具，新版的nodejs已经集成了npm

# 查看node版本：node --version

# 查看npm版本：npm -v

【安装npm和cnpm】

# 安装最新版本npm：npm install npm@latest –g/ npm install npm –g

# 安装最新测试版本npm：npm install npm@next –g

# 安装淘宝镜像cnpm：npm install -g cnpm --registry=https://registry.npm.taobao.org

【package.json】

package.json列出所有项目的依赖包，npm install会根据该文件去安装相关版本的依赖包

# 创建一个package.json：npm init

# 生成默认值的：npm init -y / npm init –yes

【dependencies和devDependencies】

"dependencies"：这些软件包是项目在生产环境中所需的，即编译后的文件也依赖的。

"devDependencies"：这些软件包仅用于开发环境，编译后不需要的，像webpack、sass

这两个对象中任何一个都只能存在唯一版本的依赖包，会发生覆盖，

但是，这两个对象的可以同时存在同一种的依赖包，也允许不同版本

# 添加一个依赖包到你package.json的dependencies：--save

# 添加一个依赖包到你package.json的devDependencies：--save-dve

【全局安装与本地安装】

本地包安装在项目目录下的node\_modules文件夹，可以直接通过require()的方式引入。

全局包安装在Node安装目录下的node\_modules文件夹，输入npm root –g查看，全局包提供的命令全局可用，不局限路径。

一般安装在全局的都是有命令行需求的包，像npm、cnpm等，但是具体项目中依赖的包需要本地安装。

# 本地：npm install <Module Name>

# 全局：npm install <Module Name> -g/ npm install <Module Name> -global

# 安装确切版本依赖包：npm install <Module Name>@<包版本号>

【删除依赖包】

删完包，node\_modules依然存在该包，需要重新生成node\_modules

# npm uninstall <package> 可以同时把dependencies和devDependencies的依赖包同时删了

【n模块】

升级依赖包需要安装n模块，n模块是专门用来管理nodejs的版本

# 安装n模块：npm install -g n

【查看模块的版本号】

# npm list <Module Name>

# 强制符： –force

# 清除npm缓存：npm cache clean