**准备工作：**

首先安装nodeJs,安装稳定版

版本查看：node -v ; npm -v

**关于npm包管理器**

简介：

npm 5+版本，所有的依赖都实现扁平化管理，在包下载的时候都会生成package-lock.json文件（npm5之前版本不会生成此文件），作用是锁定依赖安装结构，保证在任意机器上执行npm i 都会得到完全相同的node\_modules结构，因为package-lock.json存储了所有的安装信息。如果手动更新了package.json文件，执行安装命令会下载对应的新版本,并且会自动更新lock文件~

在项目中使用前：

首先在项目目录生成配置文件package.json（它保存着项目依赖信息）---执行命令：npm init

npm包全局安装位置查看：npm config edit; 配置字段：prefix=C:\node ；或者npm root -g

npm config set prefix “D:\npm\_global\_modules\node\_modules” 修改默认安装目录

npm cache clean --force：清除~/.npm的缓存(本地NPM缓存路径)目录

一些常用命令：

下载：

npm install xxx ; install可缩写为i; #为或;安装指定版本 xxx@1.1 @后面跟版本号

npm i -g xxx # -g安装到全局

npm i xxx 按装到项目（node\_modules）目录下。

npm i -save xxx # --save # -S将模块安装到项目目录下，并写入package文件的dependencies依赖项

npm i -save-dev xxx # --save-dev # -D安装到项目目录下，并在package文件devDependencies对象写入依赖

依赖详解：

devDependencies开发依赖：开发时需要的依赖，安装方式 --save-dev

dependencies 项目依赖：项目运行时必备的依赖 安装方式 --save

区别：生产环境使用npm i --production只安装dependencies部分依赖模块，开发环境npm i安装所有依赖模块（devDependencies、dependencies ）

下载命令详解：

npm i -g xxx # -g

1. 安装模块到全局，不会在项目node\_modules目录中保存模块包。

2. 不会将模块依赖写入devDependencies或dependencies 节点。

3. 运行 npm install 初始化项目时不会下载模块。

npm i xxx 但是一般我们在项目中不写依赖的用途，会被默认为--save，也就是此包，在开发和线上都会用到

1. 安装模块到项目node\_modules目录下。

2. 不会将模块依赖写入devDependencies或dependencies 节点。

3. 运行 npm install 初始化项目时不会下载模块。

npm i -save xxx # --save # -S

1. 安装模块到项目node\_modules目录下。

2. 会将模块依赖写入dependencies 节点。

3. 运行 npm install 初始化项目时，会将模块下载到项目目录下。

4. 运行npm install --production或者注明NODE\_ENV变量值为production时，会自动下载模块到node\_modules目录中。

npm i -save-dev xxx # --save-dev # -D

1. 安装模块到项目node\_modules目录下。

2. 会将模块依赖写入devDependencies 节点。

3. 运行 npm install 初始化项目时，会将模块下载到项目目录下。

4. 运行npm install --production或者注明NODE\_ENV变量值为production时，不会自动下载模块到node\_modules目录中。

升级包到最新版本：npm i xxx@latest 关键位latest

卸载安装包：只是把install 换为 uninstall ，如果带有依赖注册，则会删除注册信息； 例： npm uninstall xxx -S # -D会删除对应配置文件

检测哪些包有最新更新：首先npm install -g npm -check，后npm-check

更新安装包：1.npm i -g npm-upgrade 2. npm-upgrade

更新全局包：npm update -g

更新生成环境依赖包：npm update --save

依赖包信息查看命令：

npm info xxx 查看包详细信息

npm view xxx versions 查看包存在的所有版本号

npm view xxx version 查看最新版本

npm list 查看已经安装了哪些包

npm root -g 查看全局安装包的存放位置

npm ls xxx 查看当前安装包的版本

npm -l 查看所有命令

npm config set registry http... 更改下载镜像

npm版本管理简单了解

npm采用了semver规范作为依赖版本管理方案。semver 约定一个包的版本号必须包含3个数字

MAJOR.MINOR.PATCH 意思是 主版本号.小版本号.修订版本号

1.MAJOR 对应大的版本号迭代，做了不兼容旧版的修改时要更新 MAJOR 版本号

2.MINOR 对应小版本迭代，发生兼容旧版API的修改或功能更新时，更新MINOR版本号

3.PATCH 对应修订版本号，一般针对修复 BUG 的版本号

