1. npm全称是Node Package(包) Manager(管理器)。可以通过npm去下载几乎所有的前端工具，例如jQuery, bootstrap等。同时也是Node.js在发展过程中选择的内置包管理器, 现在Node.js已经内置npm安装包。

2. NPM的官网可以搜索所有的NPM所存储的前端包。

3. NPM管理流程：

--通过服务器作为代码仓库（registry），在里面放所有需要被共享的代码

--发邮件通知，例如 jQuery、Bootstrap、Underscore 作者使用 npm publish 把代码提交到 registry 上，分别取名 jquery、bootstrap 和 underscore。

--使用npm CLI进行安装模块等命令，npm 就会帮他们下载代码

下载完的代码出现在 node\_modules 目录里，可以随意使用了。

4. NPM CLI命令：

npm -v: 测试是否安装npm，以及显示安装的版本号。

npm install npm -g: 升级npm版本

npm install <模块名>: 本地安装模块

npm install <模块名> **-g**: 全局安装模块

npm list -g: 展示npm所安装的全局模块

npm uninstall <模块名>: 卸载npm包

npm update <模块名>: 更新制定的模块

npm help <command>: 查询某条npm命令

5.

**npm安装分为本地安装和全局安装**

本地安装

将安装包放在 ./node\_modules 下（运行 npm 命令时所在的目录），如果没有 node\_modules 目录，会在当前执行 npm 命令的目录下生成 node\_modules 目录。

可以通过 require() 来引入本地安装的包。

全局安装

将安装包放在 /usr/local/node\_modules 下或者你 node 的安装目录。

可以直接在命令行里使用。

6.

在下载安装的包中含有文件package.json， 里面含有的信息是对这个包的描述。

其中的属性包括：

name - 包名。

version - 包的版本号。

description - 包的描述。

homepage - 包的官网 url 。

author - 包的作者姓名。

contributors - 包的其他贡献者姓名。

dependencies - 依赖包列表。如果依赖包没有安装，npm 会自动将依赖包安装在 node\_module 目录下。

repository - 包代码存放的地方的类型，可以是 git 或 svn，git 可在 Github 上。

main - main 字段指定了程序的主入口文件，require('moduleName') 就会加载这个文件。这个字段的默认值是模块根目录下面的 index.js。

keywords - 关键字.

"private": true如果在package.json中设置，那么npm将拒绝发布它。目的是避免在开发过程中意外发布出去。

scripts：指定运行脚本命令的npm命令行的缩写。如：

"scripts":{

   "start": "node index.js", //运行时使用：$npm run start，就相当于执行了node index.js

   "test": "tap test/\*.js" //运行时使用：$npm run test

   "prepublish": "npm run test && npm run start",// $npm run prepublish 相当于执行了npm run test和npm run start两条命令

    }

7.

我们还可以使用npm去上传一个模块。由于每个模块都包含package.json包的描述信息，所以我们上传一个模块的时候，也需要为这个模块生成一个package.json. 使用npm的交互式问答，可以为我们生成一个大致的package.json.

然后npm adduser添加自己成为npm资源库的用户， 然后npm publish进行发布。

8.

我们在使用 npm install 安装模块的时候 ，一般会使用下面这npm命令形式：

npm init # 如果package.json不存在时，自动生成package.json

npm install #根据目录下的package.json来安装需要的模块.

如果我们安装过一次npm install后，修改package.json里面依赖的版本号后，再npm install则将会覆盖安装。

如果我们安装过一次npm install后，修改package.json里面, 去除某个依赖后，再npm install则将会卸载安装。

npm install指令执行后，会报出相应警告，例如我们警告我们安装的某个包需要另一个包的高级版本等信息。

npm install moduleName # 安装模块到项目目录下

npm install -g moduleName # -g 的意思是将模块安装到全局，具体安装到磁盘哪个位置，要看 npm config prefix 的位置。npm默认都会安装在当前的项目目录下，除非有-g标志才会安装到全局。

npm install -save moduleName # -save 的意思是将模块安装到项目目录下，并在package文件的dependencies节点写入依赖。

npm install -save-dev moduleName # -save-dev 的意思是将模块安装到项目目录下，并在package文件的devDependencies节点写入依赖。

npm install moduleName 命令

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 install -g moduleName 命令

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 install -save moduleName 命令

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(packge.json)。  
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 install -save-dev moduleName 命令

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

9.

使用npm 查看版本，我们可以使用npm查看服务器上的版本以及查看本地的版本

npm view 包 version //展示包在服务器上的最新版本以及维护的作者等信息

npm ls 包 //展示包在本地安装, 就是在当前运行npm命令的目录下的版本

npm ls 包 -g //展示包在本地全局安装的版本

10.

npm 从5.2版开始，增加了 npx 命令。当npx运行某个命令的时候，会到node\_modules/.bin路径和环境变量$PATH里面，检查命令是否存在。

11.

npm run build # 对项目进行打包，然后输出文件到dist文件夹中，使用http服务器host这些文件便可

12.

Npm可以通过npm config来对npm以key-value的形式进行配置。

例如设置npm的proxy:

Npm config set proxy [proxy地址]