NPM

介绍

全称 : Node Package Manager (Node 的包管理器),就是一个软件

包是什么

Node.js当中的包基本遵循CommonJS规范,将一组相同的模块组合在一起,形成一个完成的工具.

作用

通过NPM管理器对想要的工具包进行搜索 下载 安装 删除 发布 . 借助别人写好的包 , 可以让我们的开发变得更加便捷.

安装

安装完成Node.js那一瞬间npm自动安装了

查看npm的版本

```
npm -v
```

初始化

```
npm init     #当前文件夹的名称不能是中文
npm init -y  #建议
```

在包的制定位置会生成package.json的文件

```
{
  "name": "NumJs",          //包的名称
  "version": "0.0.0",       //包的版本
  "description": ";laarg",      //包的描述
  "main": "index.js",           //入口文件
  "scripts": {                 //脚本的配置
    "test": "echo \"Error: no test specified\" &&
exit 1"
  },
  "author": "Hal",             //作者
  "license": "ISC"            //版权生命
}
```

注意:包的名称不能中文不能大写,不能使用npm作为包名

npm init -y 一般运行在你项目的root目录的当中,和.git文件方一个层级

搜索包

一般包的搜索需要在官网进行,地址https://npmjs.org

安装工具包

```
npm install jquery
npm i jquery
npm i lodash

#安装package.json (dependencies 用于生成环境的)
npm install jquery --save
npm install jquery -S

#安装package.json (devDependencies 用于开发环境的)
npm install babel --save-dev
npm install babel -D
```

```
#npm包安装成功以后会生成package-lock.json文件
#文件锁 : 记录文件的版本的
#一般在git当中,npm的包的内容(node_modules)是不上传
#package.json是需要上传到git的,只要拿到package.json文件,那么别人就可以恢复报的内容

#你的同时拿到package.json 需要执行 npm i自动下载

#当前文件当中没有package.json,一定不要使用npm下载
```

全局安装

安装的包不在某个指定的局部文件夹里面

安装在当前的系统中

```
#less编译器
npm install less -g
#lessc .\demo.less .\demo.css

#nodemon服务器启动管理的
npm install nodemon -g
```

`全局路径 : C:\Users\用户名\AppData\Roaming\npm`

移除包

```
npm remove packname
```

依赖自动安装

你的队友根据package.json输入安装

你在安装某一个包的时候,下载了好几个包

```
比如bootstrap
    - bootstrap.css
    - bootstrap.js 依赖 jquery
```

使用流程

团队开发的流程

1.从仓库拉取代码

2.运行 npm i 安装相关依赖

3.运行项目，迭代开发

环境变量问题

1.环境变量实际上是使用系统命令控制应用的

2.每个系统当中都有环境变量的配置

win : 右键(此电脑) ->高级系统设置->环境变量

```
- 用户变量 ： 配置项仅仅只正对当前用户起到作用
- 系统变量 ： 配置作用所有用户
- 找到应用的安装目录,如果有bin/就一直写到bin目录
```

mac OS:

- VMware12-15
- https://mackext.com/

  ```
  # 系统级别
  vim /etc/profile

  #用户变量
  vim ~/.bashrc
  ```

- webstrom : idea工具换壳的编译工具

封装NPM包

首先创建一个属于自己npm项目

- npm官网注册账号,完成邮箱验证激活
- 命令行下面 : 输入 npm login
- 命令行下面 : 输入npm publish

```
#1.先创建空包
#2.npm init -y 初始化包
#3.定义文件夹
    #-定义文件
#4.设置package.json当中的信息
    # - 入口文件
#5.npm login
    # 用户名
    # 密码
    # 邮箱验证码
#6.npm publish
    # 包的名称不能是中文
    # 包的名称不能重复
```

```
#提交成功以后,可以在官网上去进行搜索和下载
#引用的时候,使用的是包的名称
require("package_name");
```

下架"垃圾包"

```
npm unpublish web2202 --force
```

cnpm

cnpm是淘宝对国外npm服务器的一个完整镜像,淘宝npm镜像

- 网站地址`https://npm.taobao.org`

安装

- win : `npm install cnpm -g --registry=https://registry.npmmirror.com`
- unix :

```
alias cnpm="npm --
registry=https://registry.npmmirror.com \
--cache=$HOME/.npm/.cache/cnpm \
--disturl=https://npmmirror.com/mirrors/node \
--userconfig=$HOME/.cnpmrc"
```

使用

自动配置完成以后,使用以下命令下载

```
cnpm i lodash

#可读 不可写
#只能下载包
#提交包是不可以的
#同步服务器 : 每10分钟请求npm官方,如果有更新下载下来
#通过npm提交了一个新的包,然后用cnpm去下载(一开始找不到,过一会)
```

修改npm的源(镜像的地址)

```
#淘宝源 不建议
npm config set registry
https://registry.npmmirror.com
#npm config set registry
https://registry.npm.taobao.org

#官方源
npm config set registry https://registry.npmjs.org
```

Yarn

包管理工具,yarn是Facebook开源的包管理器,可以用来代替npm

yarn相比较npm有几个特点:

- 本地缓存,安装过包(并且没删除),再另一个项目当中安装相同的包是不会远程安装(本地拷贝);
- 并行下载 : 一次下载多个包 (npm是串行的)`npm i lodash vue`
- 精准的版本控制.保证每次安装的和上一次的一样
  - npm再安装的使用一直挑最新版本安装
  - yarn有缓存,所有可以保证每次安装的版本一致

安装

```
npm install yarn -g

#yarn i react -g 全局失效  但是可以通过配置进行改变
```

- https://classic.yarnpkg.com/en/docs#windows-stable
- win下载yarn.msi
- brew install yarn

```
yarn global add less
```

命令

- yarn --version
- yarn init -y
- yarn global add pkg 安装全局
- yarn global remove pkg 删除全局
- yarn global dir 查看全局路径
  `C:\Users\admin\AppData\Local\Yarn\Data\global`
- yarn add pkg 安装局部
- yarn add pkg --dev (npm --save-dev -D)
- yarn list 列举出已经安装过得包
- yarn (相当于 `npm i` package.json)
- yarn

```
#不建议
yarn config set registry
https://registry.npmmirror.com
```

cyarn

```
npm install cyarn -g --registry
https://registry.npmmirror.com
```

附录

指定安装版本

```
npm i jquery@1.19.0
```

npm清除缓存

```
#莫名其妙的错误
npm cache clean
```

package.json

```
1_npm
    - package.json # "name":"web2202"
    - index.js
2_npm
    - app.js  require("web2202");

#第一次报错的原因是当前2_npm当中没有package.json
#如果有package.json,那么也需要下载依赖,必须有node_module
```

```
1_npm
    - package.json
    - index.js
2_npm
    - app.js  require("../1_npm/index.js");

#require遇到了package.json优先找包,没找到包,报
错,index.js无缘解析
```

```
1_npm
    - index.js
2_npm
    - app.js  require("../1_npm/index.js");

#require导入文件
```