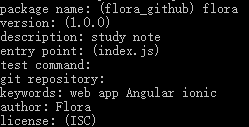
### node package manager

[npm](https://www.npmjs.com/)本来是Node.js的包管理工具，但随着JS这几年的蓬勃发展，现在的npm已经成了几乎所有跟JS相关的工具和软件包的管理工具了，并且还在不断发展完善中。**npm is the package manager for javascript.**

作用：帮助我们管理项目中所使用的库或插件，作用类似于java里面的maven。python里面的pip。

1. 安装npm
2. 跟随node一起安装
3. 进入node网站，下载长期稳定维护LTS版本，按照默认路径进行安装；安装完成之后，在Program Files文件夹里就会有nodejs文件夹。
4. 验证：node -v npm -v查看版本号
5. 单独更新：npm install npm@latest -g
6. 使用npm**创建**项目：
7. 在目标地址下打开命令行窗口，进行初始化。 有两种方式：
8. npm init -f自动填写信息,mac OS手动创建node\_modules



这里会根据输入的信息生成一个package.json的文件。

1. npm init 手动填写信息。
2. 安装项目所依赖的包：
3. npm install jquery自动安装最新版本到本地的node\_modules文件夹（安装相同的包时，旧的包会被新的包替换）
4. npm install jquery -g 安装到系统目录，地址是C: \ Users \ ciro (username) \ AppData \ Roaming \ npm \ node\_modules文件夹，注意不是在node所在的文件夹。可以用npm config list 列出全局安装时默认的地址、npm / node版本号。
5. npm install jquery --save或npm install jquery@版本号。加上**--save**的作用是可以在node modules 里的package.json dependencies里记录下包的版本号，有的不加--save也会自动记录版本号。
6. npm install jquery **–save-dev** 依赖会添加到devDependencies.

运行时需要用到的包使用--save，开发时使用的包用 -save-dev。

这时装的jquery里面会有很多东西，有些是源码，不需要关心。需要引入的只是dist里面的jquery.js，引用的时候正常引就可以了：

<script type="text/javascript" src="node\_modules/jquery/dist/jquery.js"> </script>

1. 卸载：
2. npm uninstall jquery –save
3. npm remove jquery –save，同时删除package.json内的相关信息。
4. 对于**已有**项目，如何安装所有的依赖？
5. 把package.json 拷到新项目下
6. npm install或npm install **--production** 自动下载package.json中dependencies属性对应的所有包及依赖项。
7. yarn简介

除了npm之外，还有一个包管理工具叫yarn，是facebook开发的一个node包管理工具，比npm下载依赖更快（并行下载）。

安装yarn：

npm install yarn -g

使用yarn安装项目的依赖：  
yarn

后面不需要写install，它会自动去分析依赖，看哪些需要更新。

1. browser-sync更改代码之后自动刷新浏览器，多浏览器多设备同步。需要使用npm安装:npm install browser-sync -g安装到全局。

shift + 右键可在当前文件窗口打开cmd，不需要用window + R

bowser-sync start --server --files “./fileDirectory, ./index.html, app.css, ./css/\*.css, \*/\* ”

--server是网站根目录，--files 参数指定要监视的文件，引号内跟需要监视的文件路径，多个文件使用逗号分隔。

项目目录下执行该命令行，启动服务，监听某文件的变化，自动刷新页面。

开启服务后，在浏览器内使用 control+v 打开browser-sync的控制面板。

1. gulp 前端自动化构建工具
2. gulp核心的五个方法：

gulp中是一个个task的形式来实现功能。

1. task(“task name”,function(){})
2. src(“./app.js”)
3. dest(“./minjs/”) //指定处理后的文件的输出路径。
4. watch(“./\*.js”,[“task name 1”,“task name 2”]) //监视
5. run(“task name”); //手动执行指定的任务
6. gulp的安装：使用npm，npm install -g gulp-cli 或 npm install gulp --save-dev [–save-dev]表示工具。

--save可以加到dependencies

--save-dev 加到dev下面的dependencies

1. gulp使用
2. 新建 gulpfile.js
3. gulpfile.js 文件内写入需要执行的代码   
   gulp.src(...).pipe().pipe(gulp.dest('./xxx'))通过src指定需要处理的文件，再通过pipe( … )来指定一系列任务来处理，最后使用pipe(gulp.dest('./xxx'))指定输出目录。

//先通过require()得到gulp对象  
var gulp = require('gulp');  
var press = require('gulp-uglify');  
var concat = require('gulp-concat');  
var cssnano = require('gulp-cssnano');  
var htmlmin = require('gulp-htmlmin');  
//新建任务  
gulp.task('script' , function(){ //执行script任务  
gulp.src(['./LEN.js','./L.js'])

//匹配到LEN.js , L.js 如果使用多个规则，则需要使用数组来书写第一个参数，数组中的每一个元素都是一个规则。  
.pipe(concat('all.js')) //把文件合并成all.js  
.pipe(press()) //执行什么任务(压缩)  
.pipe(gulp.dest('./lib')); //输出到指定目录  
});  
  
gulp.task('style',function(){  
gulp.src(['./css/a.css','./css/b.css'])  
.pipe(concat(all.css))  
.pipe(cssnano())  
.pipe(gulp.dest('./dist'))  
});  
gulp.task('html',function(){  
gulp.src('./index.html')  
.pipe(htmlmin({collapseWhitespace:true})) //合并空白符  
.pipe(gulp.dest('./'))  
});  
gulp.task('watch',function(){  
gulp.watch(['./LEN.js','./L.js'] , ['script']);  
});

watch监视文件变化，并调用指定任务处理代码。gulp.watch(['./LEN.js','./L.js'] , ['script']);多个参数用分别数组表示，监视LEN.js 和L.js ，若有文件发生变化，则调用script任务并执行。

1. gulp插件

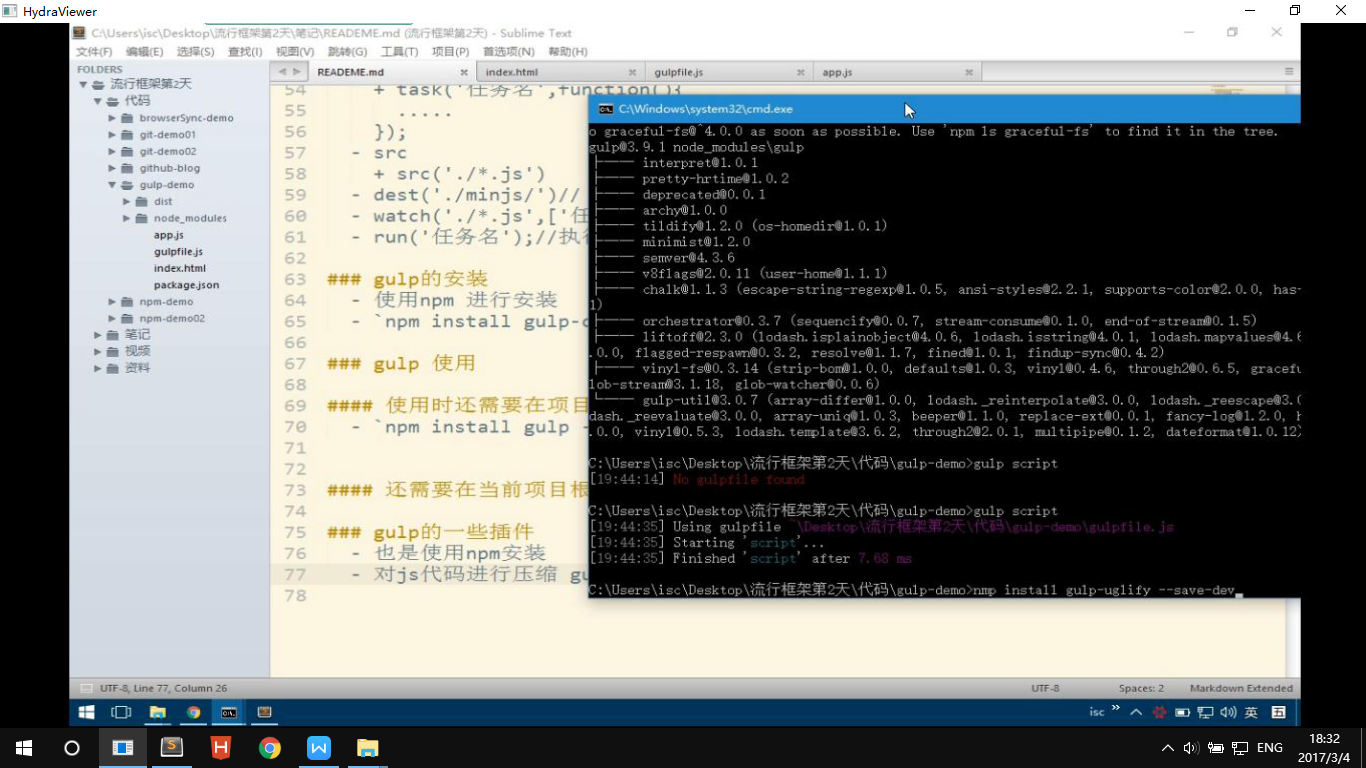
|  |  |  |  |
| --- | --- | --- | --- |
| gulp-gulify | 压缩、混淆js文件 | gulp-cssnano | 压缩css文件 |
| gulp-concat | 合并js、css文件 | gulp-htmlmin | 压缩html |

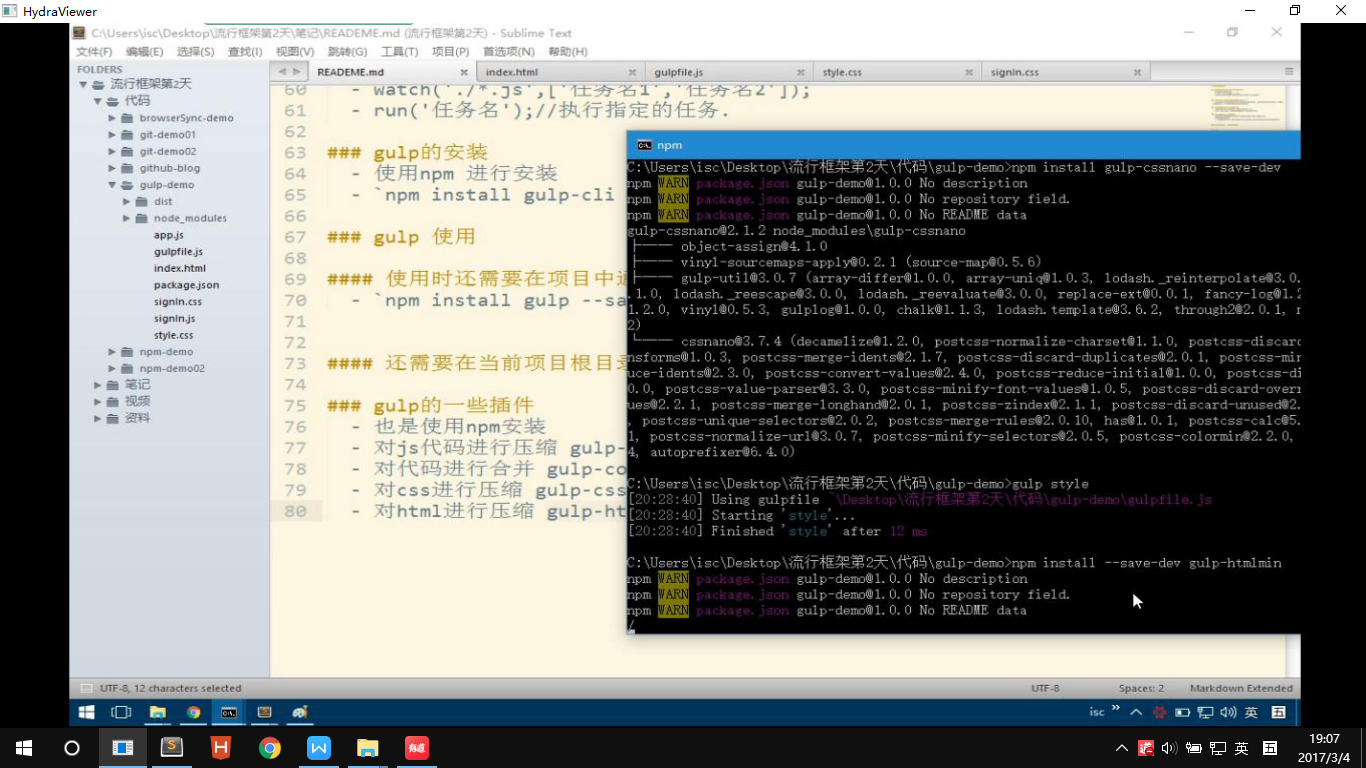
下载插件地址：npmjs.com/package/plugins

gulpjs.com/plugins

gulpjs.com.cn

先在本地的node.modules里面去搜索





browser-sync --server “./dist” --files “index.html”

#### 项目结构

自己做新项目流程

mkdir创建文件夹，构建项目整体结构

* 1. src => source源代码
     1. mkdir 创建存放源文件的文件夹(存放原始代码)
        1. resource 存放外部资源 图片等
        2. 其他每个页面新建一个文件夹(login…search…list….)
        3. touch index.html 主页
  2. dist 存放最终项目文件 ( 源代码整合压缩等处理过后 )
  3. test 测试文件存放目录
  4. docs 项目说明文档
  5. …

创建一些配置文件

* 1. touch gulpfile.js => 生成gulpfile.js文件，配置需要gulp执行的各项任务
  2. touch .gitignore => 生成.gitignore文件，配置与git服务器同步文件时需要忽略的内容 (一般只忽略 node\_modules/ 文件夹 和 .gitignore 文件)
  3. touch readme.md 创建readme.md 文件 ，用于描述项目
  4. npm init -f => 创建 package.json 文件 ，用于记录项目相关信息及项目依赖项(使用的插件、工具、库等各类包)

安装各类项目依赖项，并将项目上传至git服务器

* 1. npm install gulp --save-dev 安装gulp
  2. git commit -m 'init' 提交项目

gulp 及 browser-sync 的一些配置

* 1. 利用 gulp 监视(watch) src 文件夹内的内容，内容有改变，自动同步( 压缩、合并等处理 )到dist文件夹内
  2. 利用browser-sync监视(watch)dist文件夹内的内容，有内容改变，浏览器自动同步刷新

进公司接收现成项目

1. 拿到项目代码托管服务器的地址，将项目clone至本地。
2. 拿到项目代码后 ，现在项目文件只差 **node\_modules** 文件夹，使用npm install --production 下载项目所需依赖项(即node\_modules文件内的所有内容)
3. 下载完毕，执行gulpfile.js内的任务即可(各类watch任务)