gulp介绍

基于node流的自动化构建工具。

**gulp能干什么？**

**自动化处理一些事情**

1：自动压缩JS文件。

2：自动压缩CSS文件

3：自动合并文件。

4：自动编译sass。

5：自动刷新浏览器。

等等。

gulp网址：http://www.gulpjs.com.cn

## 工程化的作用

把一整套前端工作流程中能用工具搞定的部分，用工具搞定。也就是将前端开发流程、技术、工具、经验等规范化，标准化。

前端工程自动化工具：grunt、gulp等。

## node环境安装

**nodeJS**：Node.js是一个Javascript运行环境(runtime)。实际上它是对Google V8引擎进行了封装。V8引 擎执行Javascript的速度非常快，性能非常好。

**NPM**：全称是Node Package Manager，是随同NodeJS一起安装的包管理和分发工具，它很方便让JavaScript开发者下载、安装、上传以及管理已经安装的包。 ，它已经成为了非官方的发布Node模块（包）的标准。

nodejs自身提供了基本的模块，但是开发实际应用过程中仅仅依靠这些基本模块则还需要较多的工作。幸运的是，Nodejs库和框架为我们提供了帮助，让我们减少工作量。但是成百上千的库或者框架管理起来又很麻烦，有了NPM，可以很快的找到特定服务要使用的包，进行下载、安装以及管理已经安装的包。

**使用NPM，必须安装nodeJS。**

**npm常用命令：**

npm -v：查看npm安装的版本。

npm init：会引导你创建一个package.json文件，包括名称、版本、作者等信息。

npm list：查看当前目录下已安装的node包。

npm ls：查看当前目录下已安装的node包。

npm install moduleNames：安装Node模块到本地目录node\_modules下。

npm install <name> -g：将包安装到全局环境中。

npm install <name> --save：安装的同时，将信息写入package.json中

项目路径中如果有package.json文件时，直接使用npm install方法就可以根据dependencies配置安装所有的依赖包，这样代码提交到git时，就不用提交node\_modules这个文件夹了。

npm install <name> --save-dev：安装的同时，将信息写入package.json中项目路径中如果有package.json文件时，直接使用npm install方法就可以根据devDependencies配置安装所有的依赖包，这样代码提交到git时，就不用提交node\_modules这个文件夹了。

npm uninstall moudleName：卸载node模块。

## 4：gulp安装

因为它基于nodeJS，因此需要先安装node环境。

安装完成后，打开你的命令行窗口：输入：node -v，检测一下node是否安装好。

1-在项目里，使用npm init 命令，创建一个package.json

2-使用npm install 命令安装需要的模块

npm install moduleNames：安装Node模块到本地目录node\_modules下。

npm install <name> -g：将包安装到全局环境中。

npm install <name> --save-dev：安装的同时，将信息写入package.json中

安装 gulp

npm install gulp -g 将包安装到全局环境中。（npm i gulp -g）

npm install gulp --save-dev 安装到本地目录，同时将安装信息写入package.json中(npm i gulp -D)

注：由于网络经常不好，考虑把npm换成国内的源，可以使用nrm切换软件源。

npm install -g nrm

nrm ls　　// 查看所有的支持源（有\*号的表示当前所使用的源,以下[name]表示源的名称）

nrm use [name]　　// 将npm下载源切换成指定的源

npm install 会去查看package.json 中安装信息，自动安装需要的包

## gulp使用

**gulpfile.js文件**：这是gulp的配置文件，放在项目根目录下。

gulp的API：

**gulp.src(globs[, options])**：输出（Emits）符合所提供的匹配模式（glob）或者匹配模式的数组（array of globs）的文件。

**gulp.dest(path[, options])**：能被 pipe 进来，并且将会写文件。并且重新输出（emits）所有数据，因此你可以将它 pipe 到多个文件夹。如果某文件夹不存在，将会自动创建它。

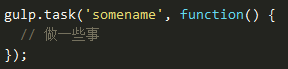
path参数：类型为String 或 Function，文件将被写入的路径（输出目录）。也可以传入一个函数，在函数中返回相应路径。

options参数：类型为Object。

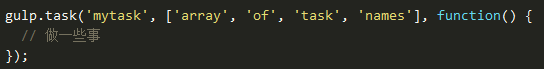
**gulp.task(name[, deps], fn)**：定义一个任务。

name参数：任务的名字，如果你需要在命令行中运行你的某些任务，那么，请不要在名字中使用空格。

fn参数：该函数定义任务所要执行的一些操作。通常来说，它会是这种形式：return gulp.src().pipe(someplugin())。



deps参数：类型为Array。一个包含任务列表的数组，这些任务会在你当前任务运行之前完成。



**gulp.watch(glob, tasks)**：监视文件，并且可以在文件发生改动时候做一些事情。它总会返回一个 EventEmitter 来发射（emit） change 事件。

glob参数：类型为String or Array。一个 glob 字符串，或者一个包含多个 glob 字符串的数组，用来指定具体监控哪些文件的变动。

tasks参数：类型为Array。需要在文件变动后执行的一个或者多个通过 gulp.task() 创建的 task 的名字。

## 

## gulp插件安装（压缩、合并、及时刷新）

https://www.npmjs.com

**gulp常用插件**

**Pump 检测js语法错误 安装 gulp i pump -D**

**gulp-uglify 压缩js代码**

**gulp-babel es6转es5**

**gulp-sass sass-》css**

**gulp-htmlmin 压缩html**

**gulp-concat 合并文件**

**使用方法，如下**

**var** gulp = require(**'gulp'**);  
  
*//gulp.src gulp.dest gulp.task gulp.watch  
  
//检测js语法***var** pump = require(**'pump'**);  
*//压缩js插件***var** uglify = require(**'gulp-uglify'**);  
  
*//获取babel插件 es6-》es5***var** babel = require(**'gulp-babel'**);

*//压缩js 的任务*gulp.task(**'compress'**, **function** (cb) {  
 pump([  
 gulp.src(**'./src/js/\*.js'**),  
 babel({**'presets'**:**'es2015'**}),  
 uglify(),  
 gulp.dest(**'./dist/js'**)  
 ],  
 cb  
 );  
});  
  
  
*// 引入gulp-ruby-sass***var** sass = require(**'gulp-ruby-sass'**);  
gulp.task(**'sass'**, () =>  
 sass(**'./src/sass/\*.scss'**,{**'style'**:**'compressed'**})  
 .on(**'error'**, sass.logError)  
 .**pipe**(gulp.dest(**'./dist/css'**))  
 .**pipe**(gulp.dest(**'./src/css'**))  
);  
  
  
*//gulp-htmlmin 压缩html***var** htmlmin = require(**'gulp-htmlmin'**);  
gulp.task(**'htmlmin'**, **function**() {  
 **return** gulp.src(**'src/html/\*.html'**)  
 .**pipe**(htmlmin({**collapseWhitespace**: **true**}))  
 .**pipe**(gulp.dest(**'dist/html'**));  
});  
  
  
*//合并js gulp-concat***var** concat = require(**'gulp-concat'**);  
gulp.task(**'concat'**,**function**(cb){  
   
 pump([  
 gulp.src([**'./src/js/a.js'**, **'./src/js/b.js'**]),  
 babel({**'presets'**:**'es2015'**}),  
 concat(**'all.js'**),  
 uglify(),  
 gulp.dest(**'./dist/js'**),  
 gulp.dest(**'./src/js'**)  
 ],  
 cb  
 );

*//刷新***var** connect = require(**'gulp-connect'**);  
  
gulp.task(**'connect'**,**function**(){  
 gulp.src(**'./src/html/\*.html'**)  
 .**pipe**(connect.reload())  
})  
  
  
*//压缩js css html 任务*gulp.task(**'build'**,[**'compress'**,**'sass'**,**'htmlmin'**],**function** () {  
 **console**.log(**'项目的构件压缩完毕！'**);  
});  
  
  
*//监听任务*gulp.task(**'watch'**,**function**(){  
 connect.server({  
 **livereload**:**true** });  
 gulp.watch(**'./src/js/\*\*/\*.js'**,[**'concat'**,**'compress'**]);  
 gulp.watch(**'./src/sass/\*.scss'**,[**'sass'**]);  
 gulp.watch(**'src/html/\*.html'**,[**'htmlmin'**,**'connect'**]);  
   
   
})