名字 模块 方法 属性 路径

gulp是基于pipe实现的

gulp安装到全局下(为了实现)

1.安装

- 全局 npm install gulp -g

- 本地 npm install gulp -D

2.创建 gulpfile.js

- gulp默认的执行文件

3.引入 gulp

- require("gulp")

4.注册(创建)任务

- gulp.task(任务名("default"),callback/()=>{})

5.执行任务(命令)

- gulp 任务名 或 gulp(需要把任务名设为default)

const gulp = require("gulp");

const gulpConcat = require("gulp-concat");

const autoprefixer = require("gulp-autoprefixer");

const gulpSass = require("gulp-sass");

const minCss = require("gulp-clean-css");

const gulpBabel = require("gulp-babel");

const uglify = require("gulp-uglify");

const htmlmin = require("gulp-htmlmin");

const webserver = require("gulp-webserver");

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

操作css的:

gulp.src()

+ 功能:

- 读取指定文件

+ 参数

- "文件路径"

gulp-sass(包)

+ 功能: 编译为css

gulp-autoprefixer(包)

+ 功能: 添加内核前缀

gulp-concat(包)

+ 功能: 合并

gulp-clear-css

+ 功能: 压缩

案例:

gulp.task("名字", () => {

return gulp.src("文件路径")

.pipe(require("gulp-sass")())

.pipe(require("gulp-autoprefixer")({

browsers: ['last 2 versions']

}))

.pipe(require("gulp-clean-css")())

.pipe(gulp.dest("写入路径"));

})

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

操作JS的:

gulp.src()

- 功能

- 读取文件

+ 参数

- "文件路径"

gulp-babel

- 功能: 读取

- 依赖:

babel-core

babel-preset-es2015

+ 版本号

^7.0.0

gulp-concat

- 功能:合并

gulp-uglify

- 压缩

gulp.dest

- 输出

案例:

gulp.task("名字", () => {

return gulp.src("文件路径")

.pipe(require("gulp-babel")({

presets: "es2015"

}))

.pipe(require("gulp-uglify")())

.pipe(gulp.dest("写入路径"));

})

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

操作html的:

gulp.src

- 读取

gulp-htmlmin

- 压缩

gulp.dest

- 输出

案例：

gulp.task("名字", () => {

return gulp.src("文件路径")

.pipe(require('gulp-htmlmin')({

removeStyleLinkTypeAttributes: true,

removeScriptTypeAttributes: true,

removeComments: true,

removeEmptyAttributes: true,

collapseBooleanAttributes: true,

collapseWhitespace: true

}))

.pipe(gulp.dest("写入根路径"));

})

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

gulp.src 读取文件

gulp.dest()

- 功能

写文件

+ 参数

- 文件路径

gulp.watch() 监听文件变化

- 功能

- 参数

文件名,文件改动时要执行任务

案例:

gulp.task("名字", () => {

gulp.watch("监听文件路径", gulp.series("需要监听的名字"))

gulp.watch("监听文件路径", gulp.series("需要监听的名字"))

gulp.watch("监听文件路径", gulp.series("需要监听的名字"))

})

gulp.task("名字", gulp.parallel(需要监听的名字, '需要监听的名字', '需要监听的名字'))

gulp.task 注册任务

"\*" 匹配所有文件 src/\*.js ==> 找指定路径下的所有文件

"!" 除去某个文件: eg:['src/\*.js',!src/a.js] ===> 找除了a.js之外的所有文件

"\*\*" 匹配0到多个文件夹 eq:"src/\*\*/\*.js" ===> 找指定路径下的所有文件夹中的js文件

"{}" 匹配类型文件

eq:"指定路径"/a,b.js ===> 找指定路径下的a.js和b.js

eq:"指定路径"/\*.{png,jpg,gif} ==> 找指定路径下所有类型的图片

proxies:[{source:/data,target}]

gulp.task('任务名',function () {

return gulp.src(文件路径)

gulp.pipe()

gulp.pipe()

gulp.pipe(gulp.dest(文件输出目录))

})

gulp.task('名字', () => {

return gulp.src("读取文件的根路径")

.pipe(require("gulp-webserver")({

port: 3000, //端口

open: true, //自动启动浏览器

livereload: true, //自动刷新

proxies: [{ //用户代理、跨域、拦截请求、转发请求

source: "接口(url)",

target: "服务器端口 接口(url)"

}]

}))

})

gulp.task('名字', () => {

return gulp.src(".")

.pipe(require("gulp-webserver")({

port: 7777, //端口

middleware: [(req, res, next) => { //服务器

console.log(req.url)

}]

}))

})

使用FS大文件读取/写入

fs.ReadStream()

- 读取大文件

- 方法

- on()

- 参数

1."data" //=> 读取

"end" //=>

2.回调函数

- 参数

- 回调函数体

fs.WriteStream()

- 写入大文件(读完才能写,有依赖关系)