

Gulp教程

编者 xiedrsz



中科软信息系统有限公司

sinosoft

目录

[简介 2](#_Toc465757441)

[1.1概述 2](#_Toc465757442)

[1.1.1 gulp优势 2](#_Toc465757443)

[1.2 Gulp的安装 2](#_Toc465757444)

[1.2.1 全局安装 2](#_Toc465757445)

[1.2.2 本地安装 2](#_Toc465757446)

[Gulp组件 2](#_Toc465757447)

[2.1与javascript有关 2](#_Toc465757448)

[2.2安装gulp组件 3](#_Toc465757449)

[Gulp基础语法 3](#_Toc465757450)

# 简介

## 1.1概述

### 1.1.1 gulp优势

* **易于使用**

通过代码优于配置的策略，Gulp 让简单的任务简单，复杂的任务可管理。

* **构建快速**

利用 Node.js 流的威力，你可以快速构建项目并减少频繁的 IO 操作。

* **插件高质**

Gulp 严格的插件指南确保插件如你期望的那样简洁高质得工作。

* **易于学习**

通过最少的 API，掌握 Gulp 毫不费力，构建工作尽在掌握：如同一系列流管道。

## 1.2 Gulp的安装

### 1.2.1 全局安装

npm install –g gulp

全局安装后可以在命令行中直接应用。

安装完后使用gulp –v 即可验证gulp 是否成功安装

### 1.2.2 本地安装

作为项目的开发依赖（devDependencies）安装：

npm install gulp –save-dev

# Gulp组件

## 2.1与javascript有关

gulp-jshint: 检查js

gulp-uglify: 压缩js

gulp-concat: 合并文件

gulp-rename: 重命名文件

## 2.2安装gulp组件

npm install gulp-jshint gulp-uglify –save-dev

# Gulp基础语法

gulp通过gulpfile文件来完成相关任务，因此项目中必须包含gulpfile.js文件。

gulp有五个方法：src、dest、task、run、watch。

1. src：指定源文件的路径
2. dest：指定处理后文件的路径
3. watch：用来监听文件的变化
4. task：指定任务
5. run：执行任务

* gulpfile.js文件实例

// 引入 gulp及组件

var gulp = require('gulp'), //基础库

imagemin = require('gulp-imagemin'), //图片压缩

sass = require('gulp-ruby-sass'), //sass

minifycss = require('gulp-minify-css'), //css压缩

jshint = require('gulp-jshint'), //js检查

uglify = require('gulp-uglify'), //js压缩

rename = require('gulp-rename'), //重命名

concat = require('gulp-concat'), //合并文件

clean = require('gulp-clean'), //清空文件夹

tinylr = require('tiny-lr'), //livereload

server = tinylr(),

port = 35729,

livereload = require('gulp-livereload'); //livereload

// HTML处理

gulp.task('html', function () {

var htmlSrc = './src/\*.html',

htmlDst = './dist/';

gulp.src(htmlSrc)

.pipe(livereload(server))

.pipe(gulp.dest(htmlDst))

});

// 样式处理

gulp.task('css', function () {

var cssSrc = './src/scss/\*.scss',

cssDst = './dist/css';

gulp.src(cssSrc)

.pipe(sass({

style: 'expanded'

}))

.pipe(gulp.dest(cssDst))

.pipe(rename({

suffix: '.min'

}))

.pipe(minifycss())

.pipe(livereload(server))

.pipe(gulp.dest(cssDst));

});

// 图片处理

gulp.task('images', function () {

var imgSrc = './src/images/\*\*/\*',

imgDst = './dist/images';

gulp.src(imgSrc)

.pipe(imagemin())

.pipe(livereload(server))

.pipe(gulp.dest(imgDst));

})

// js处理

gulp.task('js', function () {

var jsSrc = './src/js/\*.js',

jsDst = './dist/js';

gulp.src(jsSrc)

.pipe(jshint('.jshintrc'))

.pipe(jshint.reporter('default'))

.pipe(concat('main.js'))

.pipe(gulp.dest(jsDst))

.pipe(rename({

suffix: '.min'

}))

.pipe(uglify())

.pipe(livereload(server))

.pipe(gulp.dest(jsDst));

});

// 清空图片、样式、js

gulp.task('clean', function () {

gulp.src(['./dist/css', './dist/js', './dist/images'], {

read: false

})

.pipe(clean());

});

// 默认任务 清空图片、样式、js并重建 运行语句 gulp

gulp.task('default', ['clean'], function () {

gulp.start('html', 'css', 'images', 'js');

});

// 监听任务 运行语句 gulp watch

gulp.task('watch', function () {

server.listen(port, function (err) {

if (err) {

return console.log(err);

}

// 监听html

gulp.watch('./src/\*.html', function (event) {

gulp.run('html');

})

// 监听css

gulp.watch('./src/scss/\*.scss', function () {

gulp.run('css');

});

// 监听images

gulp.watch('./src/images/\*\*/\*', function () {

gulp.run('images');

});

// 监听js

gulp.watch('./src/js/\*.js', function () {

gulp.run('js');

});

});

});