**mobile\_project**

[1. 开发环境搭建 2](#_Toc4817)

[1.1. nodejs开发环境配置 2](#_Toc16684)

[1.1.1. 安装nodejs 2](#_Toc2106)

[1.1.2. 安装cnpm 2](#_Toc158)

[1.1.3. 安装yarn 2](#_Toc8950)

[1.1.4. 安装vue-cli 2](#_Toc31960)

[1.2. vscode配置 3](#_Toc16933)

[1.2.1. 安装vscode 3](#_Toc30803)

[1.2.2. 安装插件 3](#_Toc1716)

[1.2.3. 自动格式化配置 3](#_Toc16194)

[1.3. 其他配置 4](#_Toc18233)

[1.3.1. 安装谷歌vue-devtools插件 4](#_Toc17502)

[2. 项目运行 5](#_Toc20479)

[2.1. 项目拉取 5](#_Toc7518)

[2.2. 运行 5](#_Toc14383)

[2.3. 项目命令 5](#_Toc9190)

[3. 项目结构 6](#_Toc7536)

[3.1. 总体结构 6](#_Toc26416)

[3.2. src目录 7](#_Toc24964)

[3.2.1. assets 7](#_Toc15296)

[3.2.2. components 8](#_Toc8600)

[3.2.3. filters 9](#_Toc25047)

[3.2.4. interfaces 10](#_Toc7537)

[3.2.5. plugins 11](#_Toc16530)

[3.2.6. routers 12](#_Toc13461)

[3.2.7. stores 13](#_Toc25291)

[3.2.8. utils 14](#_Toc4637)

[3.2.9. views 15](#_Toc3128)

[3.2.10. main.js 16](#_Toc9186)

[3.3. 配置项 17](#_Toc12387)

[3.3.1. babel 17](#_Toc8210)

[3.3.2. postcss 17](#_Toc8806)

[3.3.3. eslint 18](#_Toc13428)

[3.3.4. vue.config 20](#_Toc19354)

[3.3.5. dll 20](#_Toc5357)

[4. 参考资料 21](#_Toc26767)

# 开发环境搭建

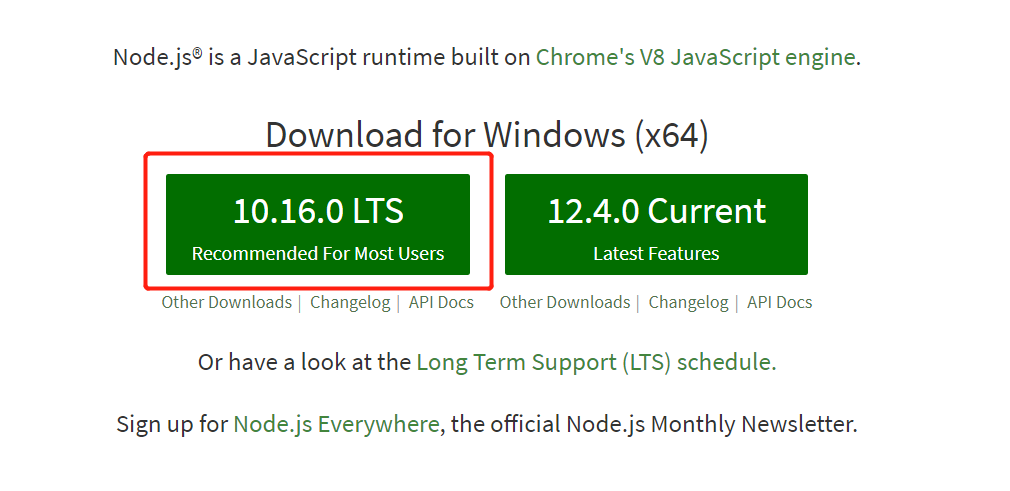
## nodejs开发环境配置

### 安装nodejs

nodejs下载地址：

<https://nodejs.org/en/>

推荐下载LTS版本，比较稳定



### 安装cnpm

安装完nodejs后，运行

npm install -g cnpm

### 安装yarn

cnpm install -g yarn

### 安装vue-cli

cnpm install -g @vue/cli

## vscode配置

### 安装vscode

vscode下载地址

<https://code.visualstudio.com/>

### 安装插件

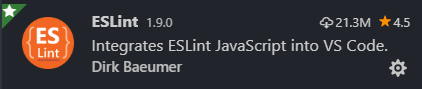
推荐安装以下几款插件

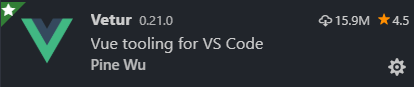
* Auto Close Tag
* Auto Rename Tag
* Chinese (Simplified) Language Pack for Visual Studio Code
* ESLint
* GitLens — Git supercharged
* Guides
* HTML CSS Support
* Import Cost
* JavaScript (ES6) code snippets
* Live Server
* npm Intellisense
* Path Intellisense
* Sass
* Vetur
* Vue 2 Snippets

### 自动格式化配置

vscode配置eslint实现自动格式化(前提项目中配置了.eslintrc.js)

安装eslint, vetur插件





在vscode的配置文件setting.json中加入eslint的配置

*// eslint配置开始*

"eslint.autoFixOnSave": true,

"eslint.options": {

"extensions": [

".js",

".vue"

]

},

"eslint.validate": [

"javascript",

"html",

{

"language": "vue",

"autoFix": true

},

]

*// eslint配置结束*

## 其他配置

### 安装谷歌vue-devtools插件

因为国内谷歌被墙的原因，无法直接从插件商店下载插件，采用拉取vue-devtools的git后运行打包

<https://github.com/vuejs/vue-devtools.git>

下面是生成好的谷歌vue-devtools插件，将其拖入扩展页面进行安装

<chrome://extensions/>



# 项目运行

## 项目拉取

项目地址：

<https://github.com/ggcer/mobile_project>

## 运行

进入项目目录，运行yarn安装依赖，依赖安装完毕后运行npm run serve启动项目

## 项目命令

运行命令

npm run serve

打包命令(不同环境运行不同的打包命令)

npm run build

npm run build:dat

npm run build:uat

npm run build:prd

依赖分析(做生成的项目的文件依赖分析)

npm run report

生成dll文件

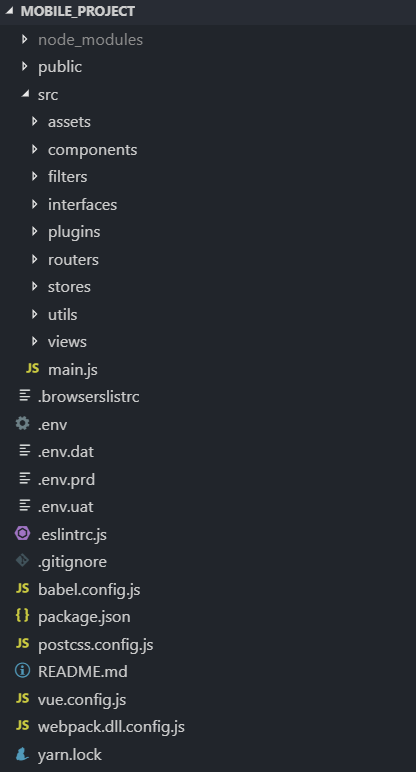
npm run dll

全项目自动格式化命令(不建议使用，最好配置vscode eslint进行项目自动格式化)

npm run lint

# 项目结构

## 总体结构



**dll配置文件**

**vue-cli配置文件**

**postcss配置文件**

**babel配置文件**

**静态文件夹**

**第三方依赖库**

**yarn.lock**

**README**

**项目依赖文件**

**git忽略配置文件**

**eslint配置文件**

**uat环境配置文件**

**prd环境配置文件**

**dat环境配置文件**

**通用环境配置文件**

**浏览器检查文件**

**入口main.js**

**页面**

**工具包**

**状态管理器**

**路由**

**vue扩展，如vue-lazyload**

**接口**

**过滤器**

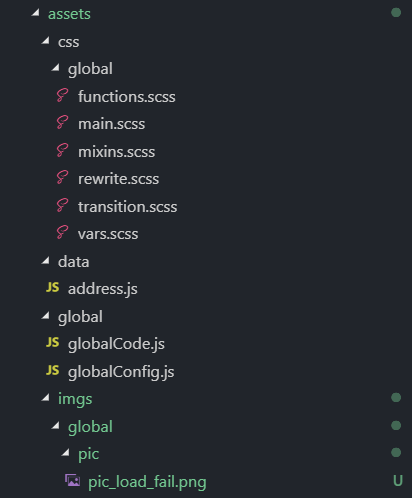
**组件库**

**静态文件夹，存放css，图片，码表等**

**主代码**

## src目录

### assets



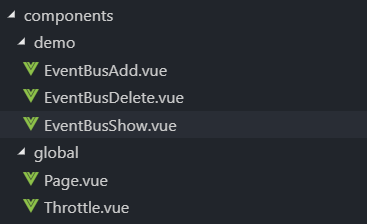
**css**：存放css / scss文件，global文件存放全局css，如main.scss，变量vars.scss等

**data**：存放数据，比如地址码表

**global**：存放全局码表**globalCode**，**globalConfig**

**imgs**：**按模块**(如user模块应该存放在user目录下)存放图片，其中**图标**存放在**icon目录**下，文件**以icon带头**，**图片**存放在**pic目录**下，文件**以pic带头，**单词之间以下划线**\_**分割

### components



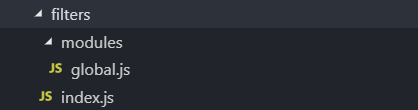
**按模块**存放组件，如Page, Throttle为全局组件，而EventBusAdd, EventBusDelete, EventBusShow为demo模块下的组件，组件命名采用帕斯卡命名，即首字母大写

**注意：组件不是越多越好，但定义组件在以下情况下很有必要**

**1：页面过于复杂，需拆分功能**

**2：公用模块，如好几个页面复用的功能模块**

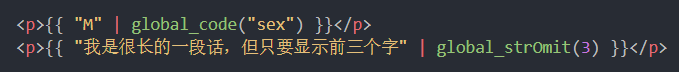
### filters



**按模块**定义过滤器，一个模块对应一个js

在使用时以管道分隔符|作为分割，使用：**模块\_filter名** 进行调用

使用：



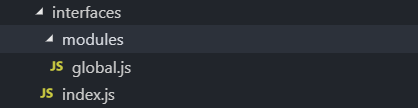
效果：



项目目前内置了一个global\_code的过滤器，可以根据globalCode.js中的码表匹配出对应的中文。

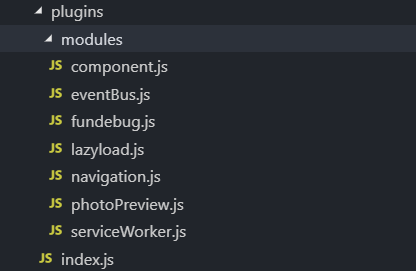
**注意：这里只存放比较通用的过滤器，比如某个模块中有多个vue使用了同一个过滤规则，则可以提出来，其他过滤器建议在各自的vue文件中定义**

### interfaces



**按模块**定义接口，一般与后台保持一致，配合http.js进行接口请求，各项目组根据实际情况进行配置

### plugins



存放vue项目的扩展，如项目需引用某个插件，则需单独定义一个js进行引入，方便维护，旨在插件的可插拔使用。

目前项目已配置的插件

component：第三方组件的引入

eventBus：可自动off的组件间通信

fundebug：线上错误监控

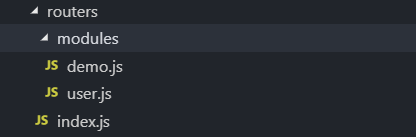
lazyload：图片懒加载

navigation：仿app的回退不刷新页面插件

ptotoPreview：图片预览(暂时未使用，有需要可开启)

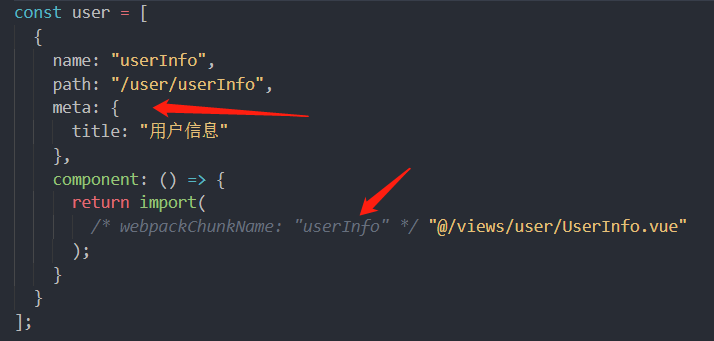
serviceWorker：pwa，vue-cli3.0自带生成

### routers



**按模块**定义页面路由(如user模块对应的路由定义文件为user.js)

定义实例：



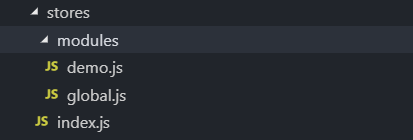
name：与页面名称一致

path：指定访问路径

meta：定义页面的元属性，如标题，可根据项目需求自行添加属性

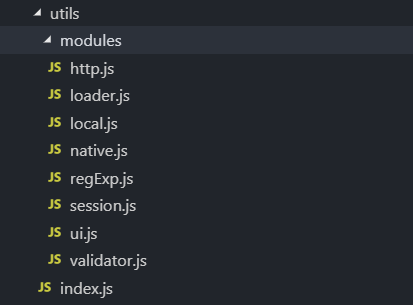
component：通过import引入对应的页面，注意其中的webpackChunkName，**相同的webpackChunkName会被打包到一块**

### stores



**按模块**定义全局状态管理器

### utils



全局的utils工具类，已经被挂载到了window对象下

目前项目已配置的工具

http：网络请求，(需根据项目前后端交互情况自行配置)

loader：加载器

local：localStorage缓存工具类

native：与原生交互的工具类，(需要原生暴露js api后配置)

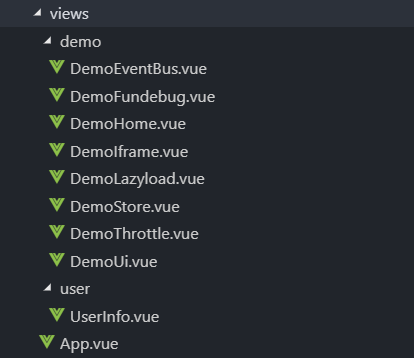
regExp：常用的正则表达式

session：sessionStorage缓存工具类

ui：ui相关工具类，如获取网页宽高等

validator：验证工具类，如身份证验证方法

### views



**按模块**定义页面(如user模块对应的页面文件夹为user)，页面命名采用帕斯卡命名，即首字母大写

### main.js



入口main.js

包含了以下内容：

1：globalCode, globalConfig, utils, interfaces全局引用，挂载到了window对象下

2：plugins引入

3：filters引入

4：全局mixin

5：根vue实例的创建，挂载router与store

## 配置项

### babel

使用vue-cli3.0本身自带的preset即可，后续如有其他特殊要求，可自行修改



### postcss

本项目配置了如下postcss插件

1：autoprefixer：css自动加浏览器前缀

2：postcss-px-to-viewport：项目本身使用vw做移动端页面适配，本身使用等同于设计稿的px值开发，渲染时会自动转换为vw进行适配，基本配置如下



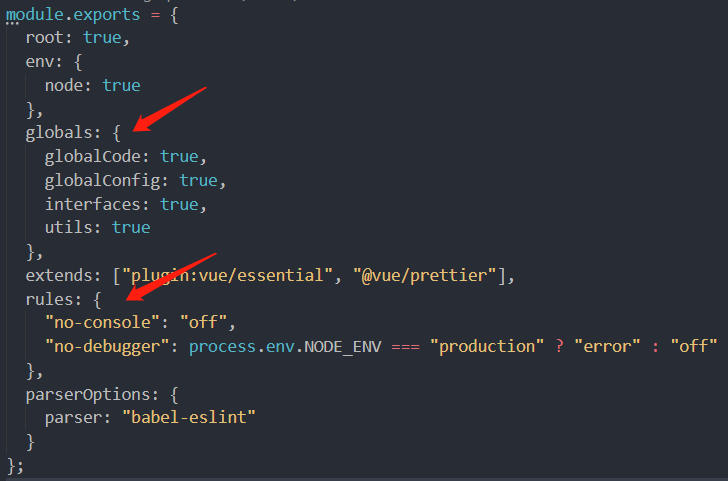
### eslint

项目使用eslint进行代码规范控制

各项目组可根据实际需求对验证规则进行调整

挂载到window中的全局对象需要配置在**globals**中，自定义的验证规则需配置在**rules**中，更多配置请查看官方文档

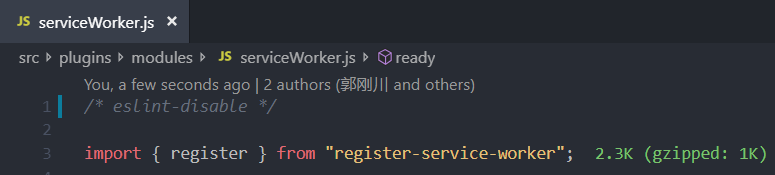
**注意重新配置规则后需重启项目**



部分特殊情况，需要在某些文件中禁用eslint检查功能，以下为常用禁用eslint检查命令

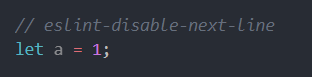
1：不检查整个文件(需放在文件头部)

/\* eslint-disable \*/



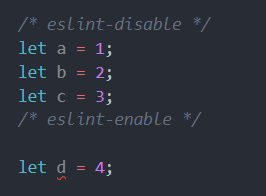
2：不检查下一行

// eslint-disable-next-line



3：不检查某个代码块

/\* eslint-disable \*/指定disable开始，/\* eslint-enable \*/指定规则校验开始



### vue.config

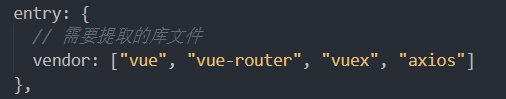
vue-cli3.0之后移除了2.0版本中的build文件夹，如果需要修改webpack相关配置，则需在vue.config.js中进行配置

### dll

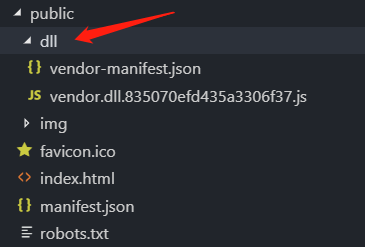
dll的作用主要是通过提取第三方库加快打包速度

其配置文件为**webpack.dll.config.js**

通过运行npm run dll即可将**entry**选项中的第三方库打包到一个dll文件夹中



生成目录为public/dll(可自行配置，但需和vue.config.js中的dll插件保持一致)



**注意：考虑到vue-devtools，开发环境默认不配置dll**

# 参考资料

vue

<https://cn.vuejs.org/v2/guide/>

vue-router

<https://router.vuejs.org/zh/>

vuex

<https://vuex.vuejs.org/zh/guide/>

vue-cli

<https://cli.vuejs.org/zh/guide/cli-service.html>

webpack

<https://www.webpackjs.com/concepts/>

scss

<https://www.jianshu.com/p/a99764ff3c41>

babel

<https://www.babeljs.cn/docs/>

postcss

<https://www.postcss.com.cn/>

eslint

<https://cn.eslint.org/>