# **vuejs**

# **VUE项目实战学习文档**

王宇轩 from BUAA

2019.07

# 目 录

[一、和简易开发的区别](#_Toc4387_WPSOffice_Level1) [1](#_Toc4387_WPSOffice_Level1)

[1.1 一个html掌握全局](#_Toc12617_WPSOffice_Level2) [1](#_Toc12617_WPSOffice_Level2)

[1.2 转为vue文件后的内容格式](#_Toc20559_WPSOffice_Level2) [1](#_Toc20559_WPSOffice_Level2)

[1. template部分](#_Toc12617_WPSOffice_Level3) [1](#_Toc12617_WPSOffice_Level3)

[2. script部分](#_Toc20559_WPSOffice_Level3) [1](#_Toc20559_WPSOffice_Level3)

[3. style部分](#_Toc9212_WPSOffice_Level3) [2](#_Toc9212_WPSOffice_Level3)

[二、 组件概念的引入](#_Toc12617_WPSOffice_Level1) [3](#_Toc12617_WPSOffice_Level1)

[2.1 主vue和小组件](#_Toc9212_WPSOffice_Level2) [3](#_Toc9212_WPSOffice_Level2)

[2.2 路由设置原理和基础设置](#_Toc30564_WPSOffice_Level2) [3](#_Toc30564_WPSOffice_Level2)

[1. 引导型“组件”](#_Toc30564_WPSOffice_Level3) [3](#_Toc30564_WPSOffice_Level3)

[2. 真正的独立组件](#_Toc12240_WPSOffice_Level3) [4](#_Toc12240_WPSOffice_Level3)

[3. 有子组件的父组件](#_Toc21712_WPSOffice_Level3) [4](#_Toc21712_WPSOffice_Level3)

[2.3 路由中引导型的设置介绍](#_Toc12240_WPSOffice_Level2) [4](#_Toc12240_WPSOffice_Level2)

[1.根目录的引导](#_Toc2162_WPSOffice_Level3) [4](#_Toc2162_WPSOffice_Level3)

[2.父组件的引导](#_Toc7510_WPSOffice_Level3) [4](#_Toc7510_WPSOffice_Level3)

[2.4 网页（组件）的跳转（切换）](#_Toc21712_WPSOffice_Level2) [5](#_Toc21712_WPSOffice_Level2)

[2.3.1 路由在组件中的形象简介](#_Toc18495_WPSOffice_Level3) [5](#_Toc18495_WPSOffice_Level3)

[2.3.2 利用路由实现组件的跳转](#_Toc24568_WPSOffice_Level3) [6](#_Toc24568_WPSOffice_Level3)

[1. 当字符串以/开头时](#_Toc4062_WPSOffice_Level3) [6](#_Toc4062_WPSOffice_Level3)

[2. 如果字符串没有以/开头纯字符串时](#_Toc6573_WPSOffice_Level3) [6](#_Toc6573_WPSOffice_Level3)

[3. 当不以/开头字符串包含/时](#_Toc29614_WPSOffice_Level3) [6](#_Toc29614_WPSOffice_Level3)

[三、组件库的使用](#_Toc20559_WPSOffice_Level1) [7](#_Toc20559_WPSOffice_Level1)

[3.1 部分vue组件库介绍](#_Toc2162_WPSOffice_Level2) [7](#_Toc2162_WPSOffice_Level2)

[3.2 引入组件库](#_Toc7510_WPSOffice_Level2) [8](#_Toc7510_WPSOffice_Level2)

[1.vant](#_Toc12307_WPSOffice_Level3) [8](#_Toc12307_WPSOffice_Level3)

[2.vuetify](#_Toc2862_WPSOffice_Level3) [9](#_Toc2862_WPSOffice_Level3)

[3.3 消除EXLint报错](#_Toc18495_WPSOffice_Level2) [11](#_Toc18495_WPSOffice_Level2)

[3.4 使用组件](#_Toc24568_WPSOffice_Level2) [13](#_Toc24568_WPSOffice_Level2)

# 一、和简易开发的区别

## 1.1 一个html掌握全局

首先是在vue项目中，抛弃了原生的html，css，js的初始样式，也就是例如meta，title和head等组件都在项目开始的一个index.html文件中进行定义，然后其余的内容均以组件（component）的形式添加，而且index.html中进行的所有操作都会适用于接下来定义的组件，例如只要该文件引入jquery，那么所有组件都可使用$()运算符。

## 1.2 转为vue文件后的内容格式

每个组件使用一个vue文件进行定义，vue推崇一组件一文件的规则，所以在一个vue组件的文件中通常包含三部分：1.template 2.script 3.style，分别是他们的html代码，js代码和css代码。

### **1. template部分**

template中和正常的html编写完全相同，只是在使用一些v指令时有些许的不同（例如v-for当item和index同时饮用一般会要求括起来），不过不影响正常运行。

### **2. script部分**

script当然也可以通过引用的方式导入，但由于语法有些差别，所以还是建议在vue文件中编写。

当引入一些组件时需要在script中使用语句import { Button } from 'vant';来进行，在该部分的开头进行定义。

然后是内部多使用export default代表在简易开发中的vue对象，在该对象内部不必有el（不必定位因为针对所有的template部分），但是对应的需要有一个name来表示当前的文件；data部分定义方法从data:{}变为了data(){return{}}，其中第二种的第二个花括号内部等同于简易开发中的花括号；在函数的定义上，除了直接函数名: function(){}之外可以直接用函数名() {}来定义函数，不清楚简易开发中是否可用，不过在vue项目中，无论是当前对象的生命钩子函数还是methods中的函数均可转化为此简便写法。

最后是在原生中可能会用到的js中定义的函数，在这里由于特殊性，推荐直接定义在export default的methods字段中，因为方便调用，而且也仅针对的当前组件使用，这样也更符合规范。

### **3. style部分**

在定义style部分后，为了保证样式只被当前组件使用，一般会在开头加一个scoped的关键字。

然后内部就是完全相同的了。

# 二、 组件概念的引入

首先根据前面的描述可知，vue的网页实际是由组件构成的，那么该部分就深入讲解下载vue项目中组件是如何和网页结合，又是如何跳转的。

## 2.1 主vue和小组件

由于index.html的存在，所以可以将vue组件填入其中显示内容，因此首先出现了一个主vue也就是在src文件夹下的App.vue，当然既然是vue类型的文件夹，它的本质也是一个组件，不过它的组件内容则是和小组件相连接的，连接方法便是使用vue自带的路由（router），通过在App.vue中内置<router-view/>的组件，便可自动根据路由的设定显示对应组件的内容。

## 2.2 路由设置原理和基础设置

路由位于src文件夹的router文件夹下，起控制作用的文件即index.js，在起内部便是将vue，路由，各组件的连接和排列。

首先开头导入vue和router，然后导入component中的小组件（名字即是export default中的name），然后在一个router对象中进行路径的设定，当然设定完毕后需要进行应用，不过在设置之前，首先为了使路由起作用，需要让Vue对象使用导入的这个路由：Vue.use(Router)即可。

设定即是对一个路径数组进行设定，因此可以先定义一个数组对象然后再利用export default new Router({数组对象})的方式来进行设定。设定方法：数组的每个元素都是一个json对象，包括内容有三种：

**1. 引导型“组件”**

仅包含path和redirect的，当访问对应path时会根据redirect指向在当前path下加上redirect内容的路径，然后根据路径显示内容；

**2. 真正的独立组件**

包括path，name，component分别是路径，组件对应的名字（起定位作用）和使用的组件，此时当进入对饮的地址，在App.vue中的router-view组件便会显示为对应的组件的内容；

**3. 有子组件的父组件**

首先当有子组件时，自身的name已经起不到作用，而且会影响子组件的显示，所以删去，此时包含path，component和children，分别是 路径，使用的组件和子组件，前两者一样，后者则是另一个json数组，包含着该父组件内router-view组件所指向的组件内容（元素类型也和这里说的router的类型一样，有三种），也就是在一个组件内部可以使用定义的其他组件，便是子组件的意义所在，也是vue组件层层包含的原理所在。

## 2.3 路由中引导型的设置介绍

当然，只是简单的添加是不行的，需要根据网址进行一定的引导：

**1.根目录的引导**

例如刚打开vue项目时会是url/#/的形式，那么此时就需要对根目录进行引导，也就是使用path: “/”,redirect: “...”，将/地址引向另一个路径对应的地址，那么注意：在数组的第一层，也就是非子组件的这一层在定义组件时是需要这样的：

路径因为是根目录所以要有/

而在根目录引导时直接即可，因为会自动加上path的/形成正确的路径。

**2.父组件的引导**

也就是对于有子组件的父组件，如何进行内部的引导，首先是它的格式如下：

外部是去除了name的正常组件，然后添加上子组件，其中子组件在定义时可以发现，path是没有/的，因为当它属于children时，他们的path前就已经有一个/了。而在引导：和根目录类似，在children中对/的路径进行引导即可，而且这种组件的引导是挺必要的，否则就会默认显示children中第一个组件，不宜维护。

## 2.4 网页（组件）的跳转（切换）

那么在设定了路由后，他们之间的跳转是如何实现的，其实就是通过对组件的$router对象进行一个push操作，也就是入栈一个新的组件来切换掉当前的组件，这里的$router即是我们在路由中设定的router数组。

### 2.3.1 路由在组件中的形象简介

这个router数组，通过this.$router便可访问，它的全部内容根据前言可知就是包含了所有路径和他们对应组件内容的数组，当然也会包含着一些其他部分；那么除了this.$router，还有一个this.$route对象，它的内容即是路由当前指定的地址，包含内容有当前的完整地址（从根目录开始的地址）和路由数组中定义的path，通过这个对象可取得当前的路径。

### 2.3.2 利用路由实现组件的跳转

跳转一般在vue文件的script部分进行，在method中定义一个方法，方法调用this.$router.push(“...”)便可实现组件的切换：

**1. 当字符串以/开头时**

直接从根目录开始将地址替换为...的内容：

，即切换App.vue中router-view的内容；

**2. 如果字符串没有以/开头纯字符串时**

切换的仅是最后一个/后的内容，也就是当前组件的子组件，即当前vue文件中template中router-view的内容：，而不会改变App.vue中router-view的内容；

**3. 当不以/开头字符串包含/时**

很显然，由于替换的当前组件中router-view的内容，所以地址的变化是将最后一个/后的内容完全替换为字符串中的内容，也就是即使：

，也会进行替换，此时即表示当前的router替换为了account组件中router-view为Controller的内容。

# 三、组件库的使用

## 3.1 部分vue组件库介绍

1. Mint-UI

饿了么团队推出，基础配色为蓝色，十分简洁较缺乏美化

网址：[https://mint-ui.github.io/#!/zh-cn](https://mint-ui.github.io/" \l "!/zh-cn)

2. WE-VUE

weui和vue结合，浓厚微信味道，基础配色为微信原生配色

网址：<https://wevue.org/>

3. Vant

有赞开发，基础配色为蓝绿（更新快，内容多，还有小程序组件库，推荐）

网址：[https://youzan.github.io/vant/#/zh-CN/intro](https://youzan.github.io/vant/" \l "/zh-CN/intro)

4. cube-UI

滴滴开发，感觉上有些劣质，基础配色为黄灰

网址：[https://didi.github.io/cube-ui/#/zh-CN](https://didi.github.io/cube-ui/" \l "/zh-CN)

5. NutUI

京东开发，商业气息较重，基础配色为红黑

网址：[http://nutui.jd.com/#/index](http://nutui.jd.com/" \l "/index)

6. Element

PC端组件库，配色丰富

网址：[https://element.eleme.io/#/zh-CN](https://element.eleme.io/" \l "/zh-CN)

7.vuetify

PC端组件库，nb就vans了

网址：<https://vuetifyjs.com/en/>

## 3.2 引入组件库

这部分将以vant和vuetify组件库为例，讲解如何在一个vue项目中使用组件库来协助页面的开发，其他组件库的安装使用只需要参考如下内容和对方的“快速上手”即可实现。

### 1.vant

1. 首先利用终端安装好vant的组件库

# 通过 npm 安装

npm i vant -S

# 通过 yarn 安装

yarn add vant

2. 引入vant的组件

这里有很多种方法引入，例如自动按需引入组件、手动按需引入组件、一次导入所有组件和通过CDN调入js的形式导入等等，这里仅介绍一个自动按需引入组件的方法（也是vant官方推荐的），剩余的方法可在vant的官网进行具体了解：

[https://youzan.github.io/vant/#/zh-CN/quickstart](https://youzan.github.io/vant/" \l "/zh-CN/quickstart)

自动按需引入组件

首先需要使用一款babel插件：babel-plugin-import，它可以再编译过程中将import的写法自动转换为按需引入的方法。

安装：npm i babel-plugin-import -D

然后在.babelrc文件中添加配置：

注意webpack 1 无需设置libraryDirectory

{

"plugins": [

["import", {

"libraryName": "vant",

"libraryDirectory": "es",

"style": true

}]

]

}

如果是babel7的用户，可在babel.config.js中配置：

module.exports = {

plugins: [

['import', {

libraryName: 'vant',

libraryDirectory: 'es',

style: true

}, 'vant']

]

};

接着只需在main.js中导入组件或者在单个的vue文件中导入即可，推荐使用前者，因为后者还需要导入Vue对象。 main.js中导入：import { Button } from 'vant'; {}中的内容根据说明文档修改为对应的内容即可。

### 2.vuetify

网址：https://vuetifyjs.com/en/（中文版翻译过于简陋，因此推荐英文版）

参考quick-start

安装：当vue-cli安装成功后，在项目根目录命令行执行vue add vuetify进行安装。

使用：有多种方法，（其实vant也有多种方法，具体可查看其官网）

下面使用最简便的方式：在src/plugins/vuetify.js中输入以下内容：

// src/plugins/vuetify.js

import Vue from 'vue'

import Vuetify from 'vuetify/lib'

Vue.use(Vuetify)

const opts = {}

export default new Vuetify(opts)

然后在main.js中添加：

import vuetify from './plugins/vuetify'

在new Vue({})中添加vuetify。

这样便可使得对于vuetify的组件和样式都可直接使用，使用后会自动导入，而不必每次使用前都use一下。

注：vuetify除了组件还包括自己的一系列样式、颜色、字体等设置，使用时只需根据vuetify左侧**Styles&animations（高级用法，非常实用）**中的内容进行class的修改即可。

下面对一些我当时难以理解的内容进行使用上的介绍：

1.v-col和v-row的使用

行和列，并且均为强制行列，在vuetify中会将v-row所在的外层框架分为12份（最近的外部组件），然后便可通过设置他们的cols来进行宽度的设定。例如对v-col设置cols=”6”则该v-col的宽度就一定是其所处组件的宽度的一半（可能会有一些间隔，即划分为12份的间隔，如果要删去这些间隔：对v-row设置no-gutters即可，即<v-row no-gutters>即可）

cols设定的是默认宽度，除了设置cols还可设置md,lg等等，具体的含义在3中介绍。

2.class中的ma,mx,mt,pb,py等等的含义

即简单的英文单词缩写，他们的值既是大小，不过是vuetify设定好的，只需要ma-2,mt-0即可。

组装：m:margin, p:padding, a:all, r:right, l:left, t:top, b:bottom, x:x轴（r和l）, y:y轴（t和b）。

使用时设定class=”ma-2”即可，不过注意一般vuetify自带的组件都会有一个默认的ma值，所以设定后会覆盖之前的内容。

3.class中的md,lg等等的含义

vuetify对屏幕监控后产生的一些对屏幕类型进行划分的标识符

lg:一般电脑屏幕

md:屏幕缩小

sm:屏幕再次缩小

xl:比一般屏幕大

因此便可利用这种类型的判断在不同类型的电脑上显示不同的内容。

display和float等等属性都可相互结合：

<https://vuetifyjs.com/en/styles/display>

<https://vuetifyjs.com/en/styles/float>

应用一：显示与不显示

添加d-md-none的class，便可让其修饰的内容在屏幕宽度缩小时消失，d即display，即在类型为md时该组件display:none。类似的还有d-none，d-lg-flex。即始终不显示和普通屏幕时显示。

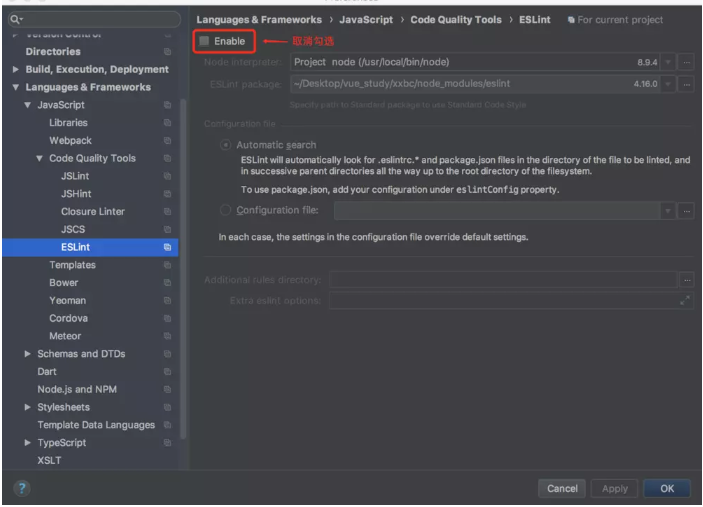
应用二：改变范围

即1中提到的设置v-col的md,lg，cols是默认，则md=””的值既是在小屏幕上该组件的显示宽度，例如cols=”6”,md=”12”，既是在普通屏幕上显示一半，在小屏幕上占满。

## 3.3 消除EXLint报错

也许有个错误提醒是很正常的，但是当添加分号甚至当缩进不对时就报错，那么就不可理喻了，因此首先关闭ESLint。

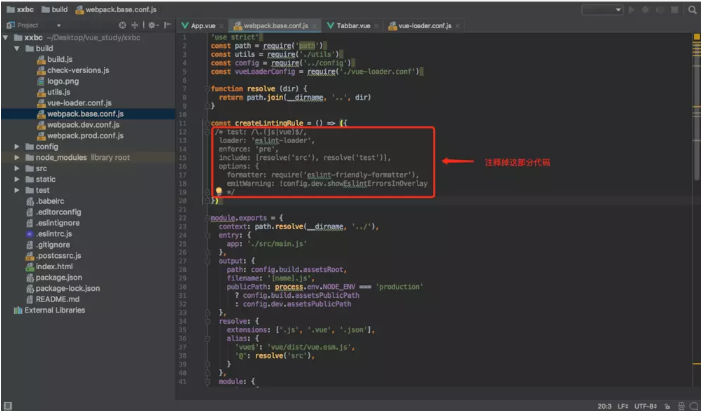
1. 在webstorm的偏好设置中关闭ESLint



2. 在vue项目的配置文件中关闭ESLint

找到build文件夹中的webpack.base.conf.js文件打开

注释以及删除掉和ESLint有关的配置



## 3.4 使用组件

在使用某个组件时，直接使用import在main.js引入后，便可在vue组件中进行使用。

添加后的组件可以添加v-show，@click等操作，也可设定id和class，同时利用.组件名的样式还可直接对当前组件内的所有该类型的组件进行控制。