当前我们company情况,使用jquery，没有使用mvc框架

1、需要提取公共js模块如common.js

2、提取公共css模块如cssreset.css，并且css作为单独文件

3、css采用scss编译

4、代码很多得不到继承公用

5、不是嵌入app的h5,大多数供浏览器解析；不需要一次性加载其他还没有用到的资源，否则会很大，启用多页面开发而非单页面开发，也是为了适应以前跳转页面刷新资源。

[**命令行生成vue2.0 -cli项目框架**](http://blog.csdn.net/zzwwjjdj1/article/details/52687488)

1. npm install -g vue-cli
2. vue init webpack vue2.0(默认安装2.0，安装这个)

vue init webpack#1.0 vue2.0(安装1.0)

1. npm install(安装依赖的东西)
2. npm run dev运行开发服务器环境
3. npm run build 生成生产环境

提醒

1、1个cli项目只有一个index.html,App.vue,main.js

2、所有组件放在：index.html中div#app里面

**文档：**

2.0官网文档：<https://vuejs.org/guide/>

2.0中文文档：<https://vuefe.cn/guide/>

**定义与使用组件步骤：**

1、定义组件，格式包括<template>,<script>,<style>

2、引入组件 import

3、注册组件 components

4、使用组件 <firstcomponent></firstcomponent>

app.vue（使用FirstComponent组件）

<template>  
 <div id="app">  
 <img src="./assets/logo.png">  
 <FirstComponent></FirstComponent>  
 </div>  
</template>  
<script>  
**import** FirstComponent **from** './components/FirstComponent'  
**export default** {  
 name: 'app',  
 components: {  
 FirstComponent  
 }  
}

或者另外一个注册组件方式

**import** homeHeader **from** '../../components/home/home-header.vue'

components: {  
 'home-header': homeHeader  
}

使用（不要与标签名字比如footer一样）

<home-header></home-header>

FirstComponent.vue（自己定义的）

<template>  
<h2>{{text}}</h2>  
</template>  
<script>  
 **export default**{  
 data () {  
 **return** {  
 text: '这是我的第一个组件'  
 }  
 }  
 }  
</script>  
<style scoped>  
 h2{  
 background: red;  
 }  
</style>

**项目使用SASS语法**

1. npm install node-sass –save-dev
2. npm install sass-loader –save-dev
3. webpack配置

{  
 test: /\.scss$/,  
 loader: "style!css!sass"  
},

1. 使用

<style lang='scss'>  
@import "./assets/scss/test.scss";//引入外部scss文件  
#app1 {  
 font-family: 'Avenir', Helvetica, Arial, sans-serif;  
 -webkit-font-smoothing: antialiased;  
 -moz-osx-font-smoothing: grayscale;  
 text-align: center;  
 color: #2c3e50;  
 margin-top: 60px;  
 & .red{  
 color:red;  
 }  
}  
</style>

1. autoprefixer的添加

webpack配置

vue: {  
 loaders: utils.cssLoaders(),  
 postcss: [  
 require('autoprefixer')({  
 browsers: ['last 2 versions','Android >= 4.0','iOS >= 7']  
 })  
 ]  
}

这个效果有用（下面代码，用的是css而不是scss）

<style scoped>  
 h2{  
 background: red;  
 display: flex;  
 }  
</style>

生成

<style type="text/css">  
 h2[data-v-34c8e960]{  
 background: red;  
 display: -webkit-box;  
 display: -webkit-flex;  
 display: -ms-flexbox;  
 display: flex;  
 }  
</style>

使用scss

<style lang='scss'>  
@import "./assets/scss/test.scss";  
#app1 {  
 font-family: 'Avenir', Helvetica, Arial, sans-serif;  
 text-align: center;  
 color: #2c3e50;  
 margin-top: 60px;  
 diplay:flex;  
 animation-name: dropAnim1;  
 & .red{  
 color:red;  
 }  
}  
</style>

生成效果

<style type="text/css">  
 html, body {  
 background: #fff;  
 -webkit-animation-name: dropAnim1;  
 animation-name: dropAnim1;  
 display: -webkit-box;  
 display: -webkit-flex;  
 display: -ms-flexbox;  
 display: flex;  
 }  
 #app1 {  
 font-family: 'Avenir', Helvetica, Arial, sans-serif;  
 text-align: center;  
 color: #2c3e50;  
 margin-top: 60px;  
 display: -webkit-box;  
 display: -webkit-flex;  
 display: -ms-flexbox;  
 display: flex;  
 -webkit-animation-name: dropAnim1;  
 animation-name: dropAnim1;  
 }  
 #app1 .red {  
 color: blue;  
 }  
</style>

解决连接：

<https://github.com/vuejs/vue-loader/issues/227>

<https://github.com/vuejs/vue-loader/issues/234>

<https://github.com/vuejs/vue-loader/issues/424>

**如何把css作为一个单独的文件，而不是以style插入html里面**

1、

**路由vue-router**

1、安装

npm install vue-router --save-dev

2、使用

在main.js中引入

**import** Vue **from** 'vue'  
**import** App **from** './App'  
**import** VueRouter **from** 'vue-router'  
Vue.use(VueRouter)  
// 定义组件, 也可以像教程之前教的方法从别的文件引入  
**const** First = { template: '<div><h2>我是第 1 个子页面</h2></div>' }  
**const** Second = { template: '<div><h2>我是第 2 个子页面</h2></div>' }  
// 创建一个路由器实例  
// 并且配置路由规则  
**const** router = **new** VueRouter({  
 mode: 'history',  
 base: \_\_dirname,  
 routes: [  
 {  
 path: '/first',  
 component: First  
 },  
 {  
 path: '/second',  
 component: Second  
 }  
 ]  
})  
// 路由器会创建一个 App 实例，并且挂载到选择符 #app 匹配的元素上。  
**new** Vue({  
 router: router,  
 render: h => h(App)  
}).$mount('#app')

App.vue

<template>  
 <div id="app">  
 <img src="./assets/logo.png">  
 <FirstComponent></FirstComponent>  
 <ul>  
 <li><router-link to="/first ">点我跳转到第一页</router-link></li>  
 <li><router-link to="/second">点我跳转到第二页</router-link></li>  
 </ul>  
 <router-view></router-view>  
 </div>  
</template>

导入css

**import** './assets/css/test.css'

**导入jquery2.2.4**

npm install [jquery@2.2.4](mailto:jquery@2.2.4) –save-dev

在index.vue引入

**import** $ **from** 'jquery'

定义没有使用的话会报错

'$' is defined but never used

**导入依赖jquery的第三方库如jQuery.md5.js**

方法一：

1. webpack.base.conf.js

**var** webpack = require("webpack")//加入

resolve: {  
 extensions: ['', '.js', '.vue'],  
 fallback: [path.join(\_\_dirname, '../node\_modules')],  
 alias: {  
 'vue$': 'vue/dist/vue',  
 'src': path.resolve(\_\_dirname, '../src'),  
 'assets': path.resolve(\_\_dirname, '../src/assets'),  
 'components': path.resolve(\_\_dirname, '../src/components'),  
 'md5': path.resolve(\_\_dirname, '../src/assets/js/lib/jquery/jQuery.md5.js'), //加入  
 'jquery':'jquery'//加入（如果是npm安装的话，可以省略）  
 }  
},

plugins: [  
 **new** webpack.ProvidePlugin({//全局暴露（否则jQuery is not defined报错），加入  
 $: "jquery",  
 jQuery: "jquery"  
 })  
],

1. index.vue使用

<template>  
 <div class="container">  
 <home-header></home-header>  
 <home-banner></home-banner>  
 <button v-on:click="addTodo" class="btn">点击</button>  
 </div>  
</template>  
<script>  
 **import** $ **from** 'jquery'  
 **import** 'md5'  
 **import** homeHeader **from** '../../components/home/home-header.vue'  
 **import** HomeBanner **from** '../../components/home/home-banner.vue'  
 **export default**{  
 data () {  
 **return** {  
 msg: 'Welcome to Your Vue.js App'  
 }  
 },  
  
 components: {  
 'home-header': homeHeader,  
 'home-banner': HomeBanner  
 },  
  
 methods: {  
 addTodo: **function** () {  
 console.log($)  
 window.alert($('.btn').get(0).tagName)  
 console.log($.md5('a'))  
 }  
  
 }  
 }  
</script>

方法二：

1、webpack.base.conf.js，loaders加入

{  
 test: require.resolve('jquery'),//添加变量  
 loader: 'expose?$!expose?jQuery!jquery'  
},

2、index.vue

**import** $ **from** 'expose?$!expose?jQuery!jquery'  
**import** 'md5'

但是报错了（因为这里用了npm安装jquery）

Module build failed: Error: jQuery requires a window with a document

解决方法：你可以在alias修改jquery指向路径

'jquery': path.resolve(\_\_dirname, '../src/assets/js/lib/jquery/jquery'),//如果是本地的话

参考地址：

<https://github.com/yiifaa/yii-template>

[http://webpack.github.io/docs/shimming-modules.html#exporting](http://webpack.github.io/docs/shimming-modules.html%23exporting)

方法三：

希望引用CDN资源，步骤如下：

1、webpack.base.conf.js

externals: {  
 wx:'jWeixin',  
 'jquery':'jQuery',  
},

2、index.vue

**import** $ **from** 'jquery'  
**import** 'md5'

**疑问，**貌似上面的配置在每个组件上都要import或者require jquery，在想有没有一种方法可以在main.js引入，其他组件就不用引入呢？

注意：不要对lib文件下进行语法检查，否则一大堆错误，npm run dev不通过，方法参照下面

**md5，base64，jquery.cookies合并成一个文件**

1、

entry: {  
 app: './src/main.js',  
 vendor:['md5','base64']  
},

2、

alias: {,  
 'base64': path.resolve(\_\_dirname, '../src/assets/js/lib/base64'),  
 'md5': path.resolve(\_\_dirname, '../src/assets/js/lib/jquery/jQuery.md5.js')  
}

之后会自动插入到html中

多个合并成一个文件导致的问题：（尚未解决）

Uncaught ReferenceError: base64 is not defined

参考：

<https://github.com/webpack/webpack/tree/master/examples/multiple-commons-chunks>

**导入第三方库flexible.js（放在项目本地），这是不依赖任何框架的。**由于ES6语法检查严格，导致整个flexible很多报错，先关了找到webpack.base.conf.js（**不建议**）

preLoaders: [  
 {  
 test: /\.vue$/,  
 loader: 'eslint',  
 include: projectRoot,  
 exclude: /node\_modules/  
 },  
 {  
 test: /\.js$/,  
 loader: 'eslint',  
 include: projectRoot,  
 exclude: /node\_modules/  
 }  
],

重启，然后在main.js加上就可以运行了

require('./assets/js/lib/flexible/flexible.js')

**这里是附加说明：**

你可以针对某个文件夹不对他进行eslint语法检查，可以在项目中找到.eslintignore后面添加这句话：

build/\*.js  
config/\*.js  
src/assets/js/lib/\*\*/\*.js

我的目录结构是这样的：



│ .babelrc

│ .editorconfig

│ .eslintignore

│ .eslintrc.js

│ .gitignore

│ index.html

│ package.json

│ README.md

├─build

│ build.js

│ dev-client.js

│ dev-server.js

│ utils.js

│ webpack.base.conf.js

│ webpack.dev.conf.js

│ webpack.prod.conf.js

│

├─config

│ dev.env.js

│ index.js

│ prod.env.js

│ test.env.js

│

├─node\_modules

│

├─note

│ byluchanan.docx

│

├─src

│ │ App.vue

│ │ filters.js

│ │ main.js

│ │ routers.js

│ │

│ ├─assets

│ │ │ logo.png

│ │ │

│ │ ├─css

│ │ │ test.css

│ │ │

│ │ ├─font

│ │ │ test.css

│ │ │

│ │ ├─images

│ │ │ └─icon

│ │ │ dot.png

│ │ │ dot\_on.png

│ │ │ search.png

│ │ │ service.png

│ │ │

│ │ ├─js

│ │ │ │ test.css

│ │ │ │

│ │ │ └─lib

│ │ │ │ base64.js

│ │ │ │

│ │ │ └─jquery

│ │ │ jquery.cookie.js

│ │ │ jquery.js

│ │ │ jQuery.md5.js

│ │ │

│ │ └─scss

│ │ │ common.scss

│ │ │ index.scss

│ │ │

│ │ └─base

│ │ \_common.scss

│ │ \_cssreset.scss

│ │ \_extend.scss

│ │ \_footer.scss

│ │ \_function.scss

│ │ \_header.scss

│ │ \_mixin.scss

│ │ \_necessary.scss

│ │ \_variable.scss

│ │

│ ├─components

│ │ │ Hello.vue

│ │ │

│ │ ├─common

│ │ │ search-icon.vue

│ │ │ service-icon.vue

│ │ │

│ │ └─home

│ │ home-banner.vue

│ │ home-header.vue

│ │ home-topic.vue

│ │

│ ├─views

│ │ └─home

│ │ index.vue

│ │

│ └─vuex

│ test.css

│

├─static

│ │ .gitkeep

│ │

│ ├─api

│ │ │ .gitkeep

│ │ │

│ │ └─Home

│ │ │ homepageV3.json

│ │ │

│ │ └─images

│ │

│ └─js

│ └─lib

│ │ base64.js

│ │

│ └─flexible

│ flexible.js

│

└─test

├─e2e

│ │ nightwatch.conf.js

│ │ runner.js

│ │

│ ├─custom-assertions

│ │ elementCount.js

│ │

│ └─specs

│ test.js

│

└─unit

│ .eslintrc

│ index.js

│ karma.conf.js

│

└─specs

Hello.spec.js

据说也可以在build/webpack.base.conf.js修改exclude: /node\_modules/改为

exclude:[/node\_modules/,/src/]貌似只在**根目录**可以（还配置alias指向'libs': path.resolve(\_\_dirname, '../libs')使用：require('libs/js/flexible.js')或者import ' libs/js/flexible.js '），在**子目录不可以（~~也许有解决方法，暂时没头绪~~，现在有解决方法了）**

exclude:[/node\_modules/,/src\/assets\/js\/lib/]，还是检查语法了，有了解决方法（**最新**）：excluede要用的是具体路径，里面不是包含正则的

**var** libs=path.resolve(\_\_dirname, '../src/assets/js/lib/')

使用

exclude: [/node\_modules/,libs]

虽然说关闭（webpack.base.conf.js注释）可以解决引入成功，但是也导致了关闭其他用ES6语法检测错误，不建议这样使用，目前**暂时（下面的方法可以关闭语法检查）**我的解决方法是把它们放到**static**文件夹下，然后在index.html引入，

<script src="static/js/lib/flexible/flexible.js"></script>

如果放在src文件下只能通过require或者import的方式来引入，如果用下面的方法，是引入不了资源的（找不到路径）！

<script src="./src/assets/js/lib/flexible/flexible.js"></script>

另外**注意**，如果不是放在static，比如远程地址的话（index.html写上下面代码），而又想在src下引入微信jssdk的话，就要这样配置，即在.vue文件中引入第三方非NPM模块。

<script src="http://res.wx.qq.com/open/js/jweixin-1.0.0.js"></script>

1、webpack.base.conf.js加上（修改这里面的东西要重新npm run dev才生效）

externals: {  
 wx:'jWeixin'  
},

jWeixin为http://res.wx.qq.com/open/js/jweixin-1.0.0.js暴露的名字，wx为import名字

2、index.html加上

<script src="http://res.wx.qq.com/open/js/jweixin-1.0.0.js"></script>

3、需要的地方使用如main.js

**import** wx **from** 'wx'  
console.log(wx)

**怎么请求后台数据并且嵌入html中**

**1、npm install vue-resource --save-dev**

**2、main.js**

**var** vueResource = require('vue-resource')  
Vue.*use*(vueResource)

**3、index.vue请求数据**

注意：模拟数据json文件要放在static并且要用get方式（POST会出现404）才能访问

**export default**{  
 data () {  
 **return** {  
 bannerList: []  
 }  
 },  
 mounted: **function** () {  
 **this**.$http.post('static/api/Home/homepageV3.json

').then((response) => {  
 // 为bannerList赋值数据，然后html就可以遍历  
 **this**.bannerList = response.data.banner\_list  
 }, (response) => {  
 console.log(response)  
 })  
 }  
}

**4、插入html**

<swipe-item class="slide" v-for="banner in bannerList"><img v-bind:src=banner.image /></swipe-item>

**vue插入style为background的代码**

v-bind:style="{ 'background-image': 'url(' + banner.image + ')' }"

使用第三方组件的参数

<swipe class="my-swipe" :auto="5000">

Class使用if进行多条件判断

之前使用artTemplate中class判断是这样的

<div class="item {{if value.data\_type==2}}destination{{/if}}{{if value.display\_type==3&&value.data\_type==1}}topic{{/if}}{{if value.display\_type==3&&value.data\_type==0}}activity{{/if}}{{if value.display\_type==2&&value.data\_type==1}}product{{/if}}">

现在使用VUE

<div v-bind:class="{'item': list.data\_type==2,  
 'item surprise':list.data\_type==1&&list.display\_type==3,  
 'item surprise activity':list.data\_type==0&&list.display\_type==3,  
 'item surprise list':list.data\_type==1&&list.display\_type==2}" v-for="list in indexData.data\_list">

参考

<http://www.blogs8.cn/posts/WcqJb2e>

项目使用vue lazyload

1、main.js

**import** VueLazyload **from** 'vue-lazyload'  
Vue.*use*(VueLazyload)

2、index.vue(line.image遍历数据)

<div class="bg cover bg\_lazyload" v-lazy:background-image="line.image"></div>

**参考资料：**

大致了解vue.js -cli

<http://www.larabase.com/collection/3/post/128>

[Vue2.0 新手完全填坑攻略——从环境搭建到发布](http://www.jianshu.com/p/5ba253651c3b)

[Vue.js 2.0 基本入门](http://www.thatyou.cn/vue-js-2-0-%E5%9F%BA%E6%9C%AC%E5%85%A5%E9%97%A8/)

[国内有哪些公司在用Vue.js，有什么心得?](https://www.zhihu.com/question/38213423)

[Vue 组件化开发实践](http://gold.xitu.io/entry/55f77eb460b28e6a6f0f4f86)

[Vue + webpack 项目实践](http://jiongks.name/blog/just-vue/)

[webpack入坑之旅（六）配合vue-router实现SPA](http://www.tuicool.com/articles/yI322iq)

[vue-router 快速入门](http://www.cnblogs.com/axl234/p/5899952.html)

[一小时包教会 —— webpack 入门指南](http://www.tuicool.com/articles/fiyQvq7)

IDE工具

<http://www.jianshu.com/p/142dae4f8b51>

e6相关

[VUE.JS组件化开发实践](http://www.cnblogs.com/rock-roll/p/5692891.html)

[Laravel 5.1 中使用 Vue2.0 组件化开发（配置）](http://www.open-open.com/lib/view/open1471589613718.html)

[vue.js2.0路由文档](http://router.vuejs.org/)

[Vue.js 开发实践：实现精巧的无限加载与分页功能](https://segmentfault.com/a/1190000005351971)

[用vue+webpack+es6+sass实现Cnode网站(一)](https://segmentfault.com/a/1190000006000118)

[在VUE2.0中使用PostCSS](http://gold.xitu.io/post/581bfc368ac247004fe174af)

[jQuery转Vue项目实践总结](http://blog.csdn.net/violetjack0808/article/details/51451672)

github源码参考

<https://github.com/825618507/vue-app>（入门适合）

<https://github.com/zerqu/qingcheng>

[webpack+vue.js重构的cnode社区的单页面(spa)应用](https://github.com/cwsjoker/Cnode-vue-spa)

[知乎日报](https://github.com/luchanan/vue-zhihu-daily)

<https://github.com/bluefox1688/vue-cli-multi-page>

<https://github.com/jiananle/vue-multi-page>

[微信高保真vuejs1.0](https://vue-wechat.github.io/)

技巧：

1、为当前目录生成tree，并且保存为tree.txt，tree命令生成文件夹，tree /f生成文件夹和文件

F:\person\_project\github\vuejs\_learn>tree /f >F:\person\_project\github\vuejs\_lea  
rn\tree.txt

2、npm run build之后如何运行项目dist

2.1、先安装

npm install -g http-server

2.2、进入打包后的项目运行，然后输入地址就好了

http-server

3、在index.vue中<style>写html,body的样式貌似没有用(不知为何)，只能写container（包含）里面的样式

<template>  
 <div class="container padding\_b\_120">  
 <home-header></home-header>

</div>

</temlate>

<style lang='scss' scoped>  
 @import "../../assets/scss/base/necessary.scss";  
 html,body{  
 background: $white;  
 }

</style>

多页面应用开发

1. webpack使用ssi include方式：使用html-loader

**import** c **from 'components/common/notice.html'**;

//上面代码会解析为字符串

<**template**>  
 <**div class="home-container"**>  
 <**h1**>{{ msg }}</**h1**>  
 </**div**>  
</**template**>  
  
<**script**> **import** c **from 'components/common/notice.html'**;  
  
 **export default** {  
 data **()** {  
 **return** {  
 **msg**: c  
 }  
 }  
 }  
</**script**>

1. 获取多页面入口路径utils.js，把所有html文件放在views文件夹下
2. exports.getEntries = **function (**globPath,type**)** {  
    **var** entries = {}  
    **var** ishtml=type!==**undefined**?**true**:**false**;  
    glob.**sync(**globPath**)**.forEach**(function (**entry**)** {  
    **if(**ishtml**)**{  
    //html以模块文件作为输出  
    //以locahost:8080/views/index.html这样形式访问  
    **var** tmp = entry.split**('/')**.splice**(**-3**)  
    var** moduleName = tmp.splice**(**0,2**)**.join**("/")**;  
    entries**[**moduleName**]** = entry  
     
    }  
    **else**{  
    //js以模块文件作为输出,比如indx.js  
    **var** basename = path.**basename(**entry, path.extname**(**entry**))**;  
    tmp = entry.split**('/')**.splice**(**-3**)**;  
    **pathname** = tmp.splice**(**1, 1**)**;  
    entries**[pathname]** = entry;  
    }  
    }**)**;  
    **return** entries;  
   }

3、<script>插入多页面html，如果htmlWebpackPlugin没有指定chunks的话，那么多所有js会插入多页面html问题

**var** pages = utils.getEntries**('./src/views/\*\*/\*.html'**,1**)**;  
**for (var** pathname **in** pages**)** {  
 // 生成html相关配置  
 **var** conf = {  
 **filename**: pathname + **'.html'**, // html文件输出路径  
 **template**: pages**[**pathname**]**, // 模板路径  
 **inject**: **true**, // js插入位置  
 **minify**: {  
 //压缩设置  
 //removeComments: true,  
 //collapseWhitespace: true,  
 //removeAttributeQuotes: true  
 // more options:  
 // https://github.com/kangax/html-minifier#options-quick-reference  
 },  
 // necessary to consistently work with multiple chunks via CommonsChunkPlugin  
 //chunksSortMode: 'dependency'  
 };  
 pathname=pathname.split**("/")[**1**]**;//去掉views  
 **if (**pathname **in** module.exports.**entry)** {  
 conf.**inject** = **'body'**;  
 //如果每个html没有进入这里的话，那么全部js将会插入html  
 conf.**chunks** = **[**pathname, **'vendor'**, **'manifest'**, **'bootstrap']**,  
 conf.**hash** = **true**;  
 }  
 module.exports.**plugins**.push**(new** HtmlWebpackPlugin**(**conf**))**;  
}