

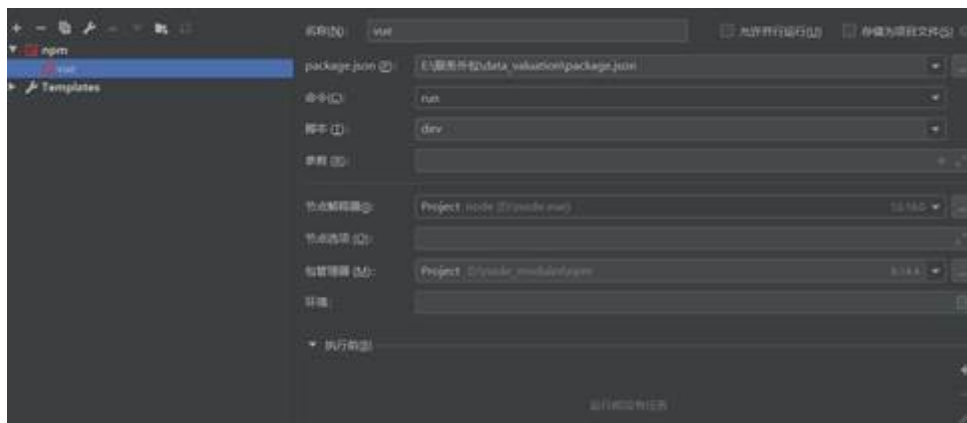
# Vue项目的实践经历

## 2.8

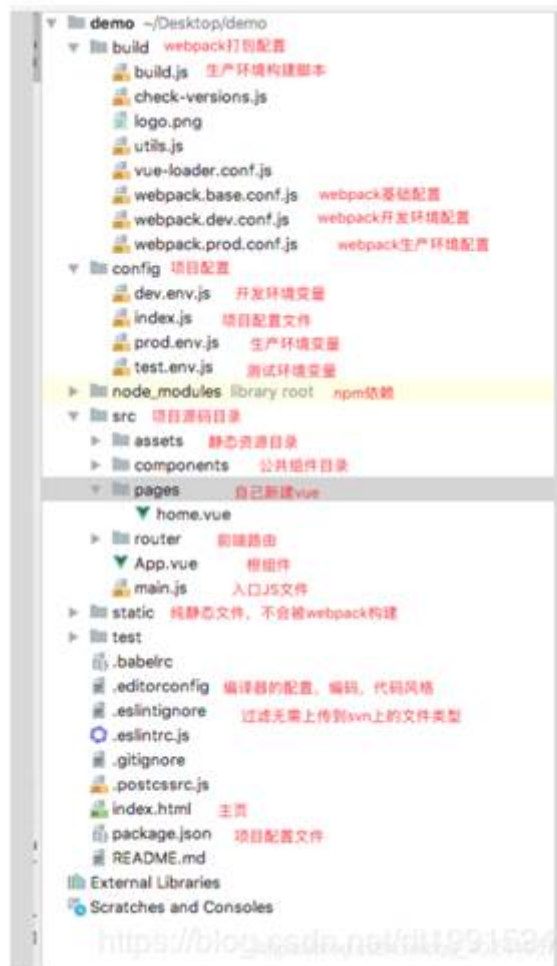


1、vue项目的搭建：  
vue项目搭建教程.docx

2、vue项目在pycharm中进行前后端分离的操作，启动指令：在终端输入：npm run dev；  
此为热部署操作，可进行动态修改。



3、vue项目的结构主要如下图所示：



4、在vue项目中引入第三方组件：axios，vuex，element-ui。

vuex：Vuex 是一个专为 Vue.js 应用程序开发的状态管理模式。它采用集中式存储管理应用的所有组件的状态，并以相应的规则保证状态以一种可预测的方式发生变化。核心是store(仓库)

5、此处的data不可省略小括号，此为固定格式。尽量多的使用插值表达式，加入将下图中的msg引入，在table下加入{{msg}}即可，data下可传输相应的数据

```

<template>
  <div>
    <table>
      <tr>
        <td>编号</td>
        <td>图书名称</td>
        <td>作者</td>
      </tr>
    </table>
  </div>
</template>

<script>
  export default {
    name: "Book",
    data(){
      return {
        msg: 'Hello Vue!'
      }
    }
  }
</script>

<style scoped>
</style>

```

6、每添加一个路由，需要在routes中加入一个新的路径，路由目录router/index.js。

7、不同于jsp中的遍历，vue使用v-for进行操作

```

</tr>
<tr v-for="item in books">
  <td>{{item.id}}</td>
  <td>{{item.name}}</td>
  <td>{{item.author}}</td>
</tr>
</table>

```

8、create函数：初始化函数，在页面一开始就会显示出来

```

@Override
public void addCorsMappings(CorsRegistry registry) {
    registry.addMapping(pathPattern: "/*")
        .allowedOrigins("*")
        .allowedMethods("GET", "HEAD", "POST",
        .allowCredentials(true)
        .maxAge(3600)
        .allowedHeaders("*");
}

```

9、解决跨越问题：

或者在控制类之上加载@crossOrigin(origins={"http://localhost:8080","null"})

10、vue项目架构

├─ build	// webpack配置文件
├─ config	// 项目打包路径
├─ dist	// 打包文件
├─ src	// 源码目录
│ └─ api	// 请求接口
│ └─ common	// 公共组件
│ └─ components	// 组件
│ └─ page	// 页面
│ │ └─ Cart	// 购物车
│ │ └─ Checkout	// 提交订单
│ │ └─ Goods	// 商品
│ │ │ └─ goods	// 商品列表
│ │ │ └─ goodsDetails	// 商品详情
│ │ └─ Home	// 主页
│ │ └─ Login	// 登陆
│ │ └─ Order	// 订单
│ │ │ └─ order	// 商品列表
│ │ │ └─ payment	// 提交订单
│ │ │ └─ paysuccess	// 提交成功
│ │ └─ User	// 个人中心
│ │ │ └─ children	
│ │ │ │ └─ addressList	// 地址列表
│ │ │ │ └─ information	// 个人信息
│ │ │ │ └─ order	// 订单列表
│ └─ index.vue	// 主页
├─ store	// vuex的状态管理
│ └─ action.js	// 配置actions
│ └─ index.js	// 引用vuex, 创建store
│ └─ modules	// store模块
│ └─ mutation-types.js	// 定义常量mutations名
│ └─ mutations.js	// 配置mutations
├─ App.vue	// 页面入口文件
├─ main.js	// 程序入口文件, 加载各种公共组件
├─ favicon.ico	// 图标
└─ index.html	// 入口html文件

## 2.9

11、在编写样式的时候出现了错误，需要使用以下命令进行样式的安装。

- 如果是 常规 的，执行 `npm install style-loader css-loader style-loader --save-dev` 安装依赖就行。
- 如果是 less 的，执行 `npm install less less-loader --save-dev` 安装依赖就行。
- 如果是 sass 的，执行 `npm install sass sass-loader --save-dev` 安装依赖就行。或者  
(`$npm intall sass-loader --save ; $npm install node-sass --save`)

12、有关element的相关知识

- `:` 代表这是一个表单
- `-> ref`: 表单被引用时的名称，标识
- `-> rules`: 表单验证规则
- `-> model`: 表单数据对象
- `-> label-width`: 表单域标签的宽度，作为 Form 直接子元素的 form-item 会继承该值
- `-> :` 表单中的每一项子元素
- `-> label`: 标签文本
- `-> prop`: 表单域 model 字段，在使用 validate、resetFields 方法的情况下，该属性是必填的
- `:` 输入框
- `-> v-model`: 绑定的表单数据对象属性
- `-> style`: 行内样式
- `-> maxlength`: 最大字符长度限制

13、在表单中加入了校验器，但是因为js和vue项目的不熟练，碰了很多的钉子，总体进度而言比较慢，因为不熟悉，仅完成了登录。在idea测试时，css的嵌套也出了很多问题，比如元素嵌套直接报错？但是已经有思路了和时间方法了，所以接下去的进度会逐渐加快

## 2.12

---

### 1、引入axios

```
import axios from 'axios'  
import VueAxios from 'vue-axios'  
Vue.use(VueAxios, axios);
```

### 2、在测试的时候用了一个json数据进行测试，测试的语句是

```
mounted(){  
  axios.get('../data.json').then(response=>(console.log(response.data)))  
}
```

### 3、目前已经完成的登录页面主要如下，后期准备加入人机验证，即拉动滑块验证

### 4、今天还主要测试了于后台的连接，主要是在浏览器的控制台中进行输出，同时连接了页面的跳转指令push，以及命令的提示输出等，接下来就要开始撰写主页面了。

## 2.13-2.14

---

### 1、主要是完成了注册页面，以及管理系统主页面的设计。对后续仍然有些模糊，注册页面目前来说还有一个人机识别的问题没有解决，需要引入相应的js文件；

### 2、在通过css样式编写的过程中，因为使用的是命令导入相应的设计，所以在某些版本上存在问题。

### 3、准备在后台界面上利用echarts，但是在排版上出现了很大的问题，仍需要进一步改进；

### 4、后续操作过程中，准备将登录页面进行修改，同时后台配合。