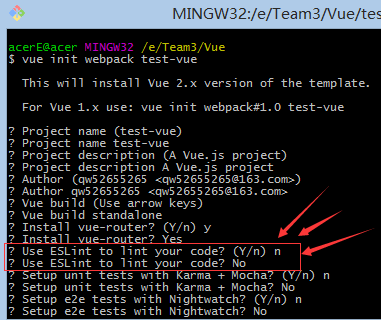
# Vue学习笔记

## 1、代码格式问题

构建项目时，ESLint选择不启用，不然，一个空格都报错，你懂的C:\Users\acerE\AppData\Local\Temp\4WWT_YLQWEJE934MRBD`77D.gif



## 2、创建项目

1、创建项目

**vue init webpack projectName（特别注意上图）**

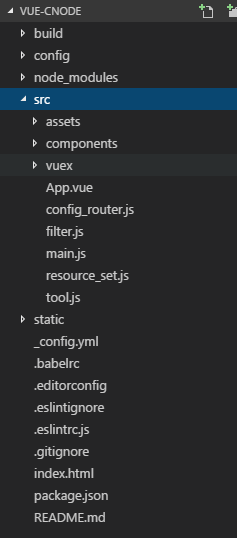
2、进入项目，下载依赖

**npm install**

3、运行项目

**npm run dev**

## 3、项目目录结构



1、build目录是一些webpack的文件，配置参数什么的，一般不用动

2、config是vue项目的基本配置文件

3、node\_modules是项目中安装的依赖模块

4、src源码文件夹，基本上文件都应该放在这里

4.1、assets 资源文件夹，里面放一些静态资源

4.2、components这里放的都是各个组件文件

4.3、App.vue App.vue组件

4.4、main.[**js**](http://lib.csdn.net/base/javascript)入口文件

5、static生成好的文件会放在这个目录下。

6、test[**测试**](http://lib.csdn.net/base/softwaretest)文件夹，测试都写在这里

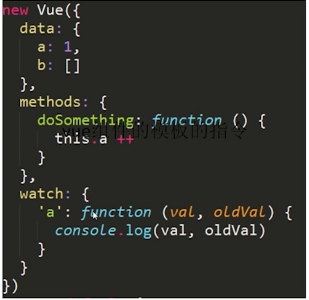
7、.babelrc babel编译参数，vue开发需要babel编译

8、.gitignore 用来过滤一些**[版本控制](http://lib.csdn.net/base/git" \t "_blank" \o "Git知识库)**的文件，比如node\_modules文件夹

9、index.html 主页

10、package.json 项目文件，记载着一些命令和依赖还有简要的项目描述信息

## 4、Vue组件重要属性



1、data：存放数据

2、methods：函数（方法）

3、watch：监听data中的数据变化

4、mounted：计算属性

5、create：组件创建

6、mounted：

7、name：名称

8、destroyed：销毁

9、props：接收父组件传入参数

## 5、模板指令

1、数据渲染：v-text,v-html,{{}}

2、控制模块隐藏：v-if(直接不加载), v-show（通过display:none隐藏）

3、渲染循环列表：v-for

4、事件绑定：v-on

5、属性绑定：v-bind

## 6、父子组件之间的通信

父组件：

给子组件传递参数：通过v-bind,可以给子组件传递参数，如下:dialogFormVisible

通过@子组件方法=”自己实现方法接收参数”的形式来实现接收子组件的通信，如下@hideDialog,（hideDialog是定义在子组件的方法）

<test-dialog

:dialogFormVisible="showDialog"

@hideDailog="handleDialog">

</test-dialog>

handleDialog(*showDialog*) {

this.showDialog = showDialog;

}

子组件：

接收父组件参数：通过props接收来自父组件的参数，（props是个数组，里面是字符串的形式,可以像定义在data中的参数一样使用）

向父组件传递参数：通过$emit方法，注册向父组件传递的方法，后面是该方法传递的参数

export default {

props: [

'dialogFormVisible'//来自父组件的参数，父组件参数更新时，子组件自动更新

]

}

closeDialog() {

this.$emit('hideDailog', false);//向父组件传值

}

# 二、Vuex学习笔记

## 1、state：单一状态树

# 三、Router：路由

路由模板： <router-view></router-view>

## 1、嵌套路由：路由参数配置children

**以 / 开头的嵌套路径会被当作根路径。**模板所渲染的组件内也可以配置路由模板。

需要在路由的参数中使用 children 配置，例如：



## 2、编程式动态路由

1、router.push(location)

这种方法可以借助 router 的实例方法，通过编写代码来实现，例如：

// 字符串

router.push('home')

// 对象

router.push({ path: 'home' })

// 命名的路由

router.push({ name: 'user', params: { userId: 123 }})

// 带查询参数，变成 /register?plan=private

router.push({ path: 'register', query: { plan: 'private' }})

2、router.replace(location)

跟 router.push 很像，唯一的不同就是，它不会向 history 添加新记录，而是跟它的方法名一样 —— 替换掉当前的 history 记录。

3、router.go(n)

这个方法的参数是一个整数，意思是在 history 记录中向前或者后退多少步，类似window.history.go(n)。例如：

// 在浏览器记录中前进一步，等同于 history.forward()

router.go(1)

// 后退一步记录，等同于 history.back()

router.go(-1)

// 前进 3 步记录

router.go(3)

// 如果 history 记录不够用，那就默默地失败呗

router.go(-100)

router.go(100)