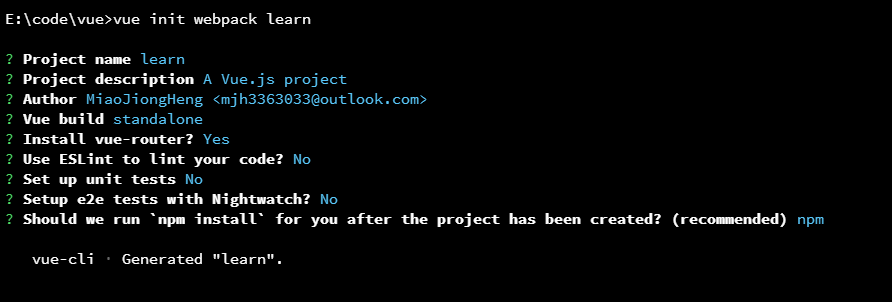
# QZ\_TL\_QD\_002 vue页面间跳转

1. 选中本地指定文件夹（指导书地址：E:\code\vue），在此处打开Hyder，输入以下代码：

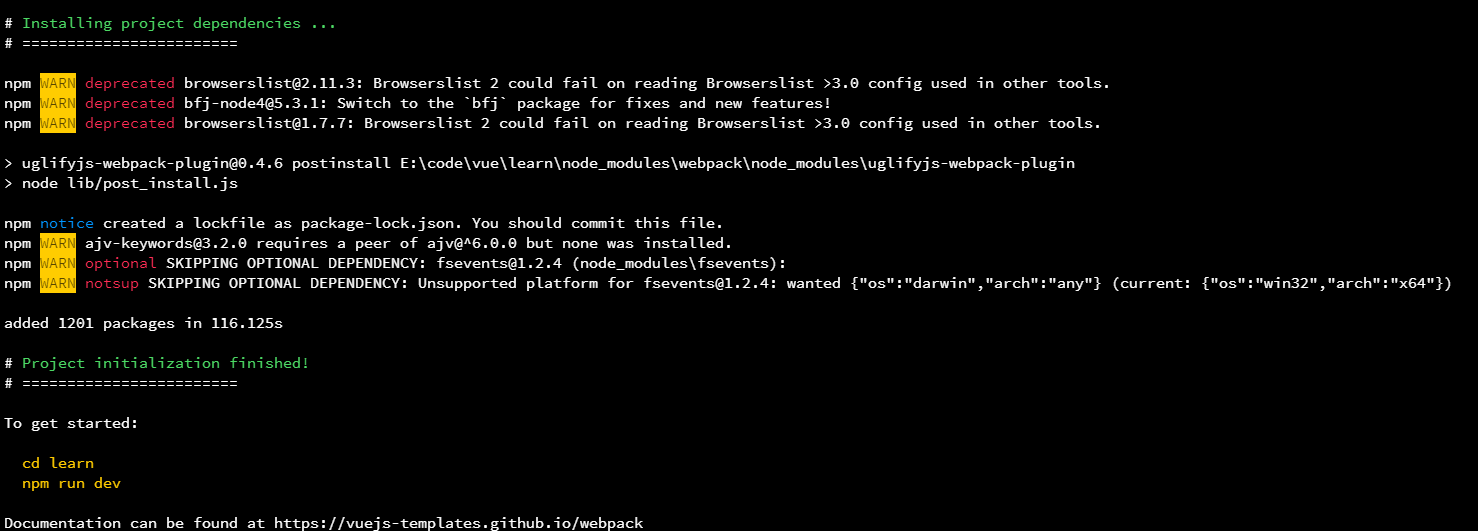
vue init webpack <project\_name>创建项目。

1. 随后会弹出一些可选项：



* ? Project description (A Vue.js project) vue-cli新建项目(项目描述)；
* ? Author (micah < mjh3363033@outlook.com >) ；mjh3363033@outlook.com(项目作者)
* ? Vue build   
  ❯ Runtime + Compiler: recommended for most users   
  Runtime-only: about 6KB lighter min+gzip, but templates (or any Vue-specific HTML) are ONLY allowed in .vue files - render functions are required elsewhere 这里选择Runtime + Compiler: recommended for most users；（直接Enter即可）
* ? Install vue-router? (Y/n) y 是否使用vue-router，这里根据需求选择；
* ? Use ESLint to lint your code? (Y/n) y 是否使用ESLint，这里根据需求选择；（一般不选择）
* ? Setup unit tests with Karma + Mocha? (Y/n) n 是否需要单元测试，这里根据需求选择；（本次开发不需要）
* Setup e2e tests with Nightwatch? (Y/n) n是否需要单元测试，这里根据需求选择；（本次开发不需要）

全部输入后：



1. 进入项目进行项目组件的下载以及项目的启动  
   cd pname(项目目录)：cd learn

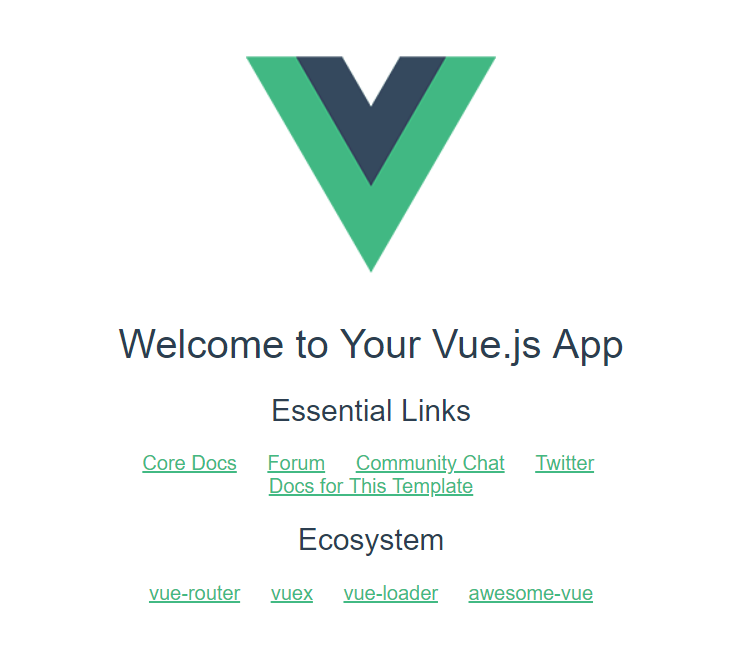
npm run dev（若速度太慢就采用cnpm run dev）



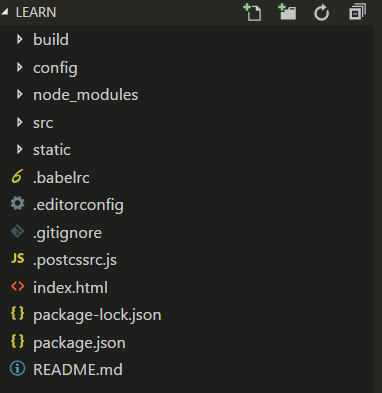
因为运行环境默认是8080端口，如果端口被占用，请在\config\index.js文件中：

将port: 8080🡺修改为任意一个没有被占用的端口后重新执行nmp run dev

1. 项目的初始界面如下图所示：



项目结构如图：



├── **build 构建脚本目录**  
│ ├── build.js 生产环境构建脚本  
│ ├── check-versions.js 版本检查脚本  
│ ├── logo.png vue logo图片  
│ ├── utils.js 构建相关工具方法  
│ ├── vue-loader.conf.js vue-loader配置文件  
│ ├── webpack.base.conf.js wabpack基础配置  
│ ├── webpack.dev.conf.js wabpack开发环境配置  
│ └── webpack.prod.conf.js wabpack生产环境配置

├── **config 项目配置**  
│ ├── dev.env.js 开发环境变量  
│ ├── index.js 项目配置文件  
│ ├── prod.env.js 生产环境变量  
│ └── test.env.js 测试环境变量

├── **src 项目源码目录**  
│ ├── main.js 入口js文件  
│ ├── app.vue 根组件  
│ ├── components 公共组件目录  
│ │ └── title.vue  
│ ├── assets 资源目录，这里的资源会被wabpack构建  
│ │ └── images  
│ │ └── logo.png  
│ ├── routes 前端路由  
│ │ └── index.js  
│ ├── store 应用级数据（state）  
│ │ └── index.js  
│ └── pages 页面目录  
│ ├── hello.vue  
│ └── notfound.vue  
├── **static 放置纯静态资源，不会被wabpack构建**  
├── **test 测试文件目录（unit&e2e）**

├── **package.json npm包配置文件，定义了项目信息与依赖包信息**  
├── **index.html 入口页面**

1. 具体的编程工作主要在src中进行。
2. Router路由的理解

用 Vue.js + Vue Router 创建单页应用，是非常简单的。使用 Vue.js ，我们已经可以通过组合组件来组成应用程序，当你要把 Vue Router 添加进来，我们需要做的是，将组件 (components) 映射到路由 (routes)，然后告诉 Vue Router 在哪里渲染它们。下面是个基本例子：

在components中创建一个tip.vue。

在router/index.js中首先引入该文件，

import Tip from '@/components/tip'

再在router:[ ]中添加

{ path: '/ Tip ',

name: ' Tip’

component: Tip}

其中path 配置了路由的路径，component 配置了映射的组件。

1. 组件跳转的实现

先在router/index.js中完成相应B组件路由的注册（详情见上一步），然后在需要完成跳转的A组件（即相应的后缀为.js的文档）上，在<template >标签中写入<router-link to="/ Tip ">显示Tip </router-link>，即可完成跳转。在<template >标签中应该写入<router-view/>来渲染通过路由通过路由映射过来的组件，当路径更改时，<router-view> 中的内容也会发生更改。简单来说，<router-view/>显示默认路由指向的组件，点击链接后完成新组件的渲染，完成跳转。。

1. 嵌套路由

创建三个vue文件，分别为Me.vue文件,Collection.vue文件,Trace.vue文件。首先注册路由，在router/index.js中用引入三个文件，语句如下：

import Me from '@/components/ Me '

import Collection from '@/components/ Collection '

import Trace from '@/components/ Trace '

再在router:[ ]中添加

{

path: '/Me',

name: 'Me',

component:Me,

children: [

{

path: 'collection',

name: 'Collection',

component: Collection

},

{

path: 'trace',

name: 'Trace',

component: Trace

}

]

}

Children[ ]中的路由即为嵌套的子路由，以“/”开头的嵌套路径会被当作根路径，所以子路由上不用加“/”;在生成路由时，主路由上的path会被自动添加到子路由之前，所以子路由上的path不用再重新声明主路由上的path了。

在Me.vue文件中的<template>标签内的<div class="tabs">块内再定义两个<div>子块，分别定义为<div class="tabs">和<div class="content">，tabs块用于显示链接，指向相应的子路由：

<router-link to="/Me/collection" tag="li" >我的收藏</router-link>

<router-link to="/Me/trace" tag="li">我的足迹</router-link>

content块用来渲染相应子路由指向的vue文件：<router-view></router-view>。

1. 自定义标签

自定义标签有两种方法，分别是在页面级组件APP.js中进行局部定义和在入口js文件main.js中进行全局定义。两种方法相比较而言，局部定义所用的计算时间和渲染时间更少。下面将简单介绍这两种方法：

局部定义：在<script>标签中写入语句import Foot from '@/components/foot'通过绝对路径查找foot.vue文件，并将其实例化为了一个Foot对象，但是没有进行加载。再在export default{ }中的component{ }中写入Foot对象，创建了Foot标签，或者通过语句Vue.component('Foot', Foot)完成标签的创建，此时该标签可作为组件显示。

全局定义：在main.js文件中写入语句import Foot from './components/foot'将foot.vue文件实例化为一个Foot对象，再使用Vue.component('Foot', Foot)创建一个名为Foot的标签，Foot对象对应的组件可通过Foot标签显示。

使用其他UI

https://blog.csdn.net/zhuming3834/article/details/72778249