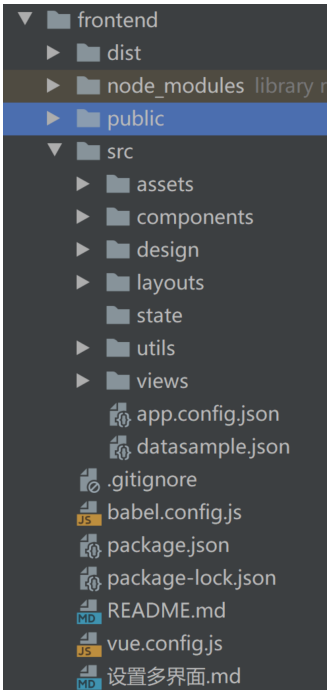
# Vue多页面

## 文件结构

### 基本框架



文件结构示例

`dist`: build之后生成的文件，每个界面的`html`都在里面

`public`: 单页面的`html`模板都会放在这里

`src`: 最重要的东西，所有的页面主要内容都在里面

`assets`: 主要是一些字体、图片之类的

`components`: 在组建页面的时候调用的`vue`组件（建议一些通用的元素可以自己写成`vue`组件，我就自己写了 一个`custombutton`）

`state`: 用来储存需要在界面间交换的参数信息（暂未实现，之后补充）

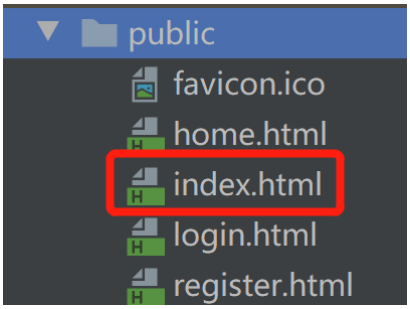
`index.js`: 这里头有个`store`的结构，我暂时没有研究怎么用，应该是跟`router`一样的方法在`main.js`上引用一下，然后在`new vue`里头声明一下，然后到`$name.vue`文件里头用`$store`就可以引用了，有需要的话可以研究一下

`views`: 存放已经实现的单页面，内部的每个文件夹都代表了一个页面

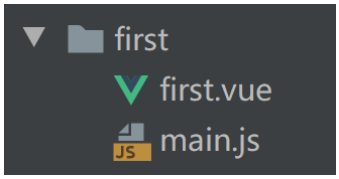
`vue.config.js`: 用来配置页面入口

### 没有内部跳转的单页面

没有内部跳转的页面的基本文件结构如下:



文件结构示例1



文件结构示例2

一共是三个文件：

- `$name.html`：放在`frontend/public`文件夹下
  - 以页面的主要功能命名的（方便查看）`html`文件，内部包含了一个最基本的模板
  - `index.html`是首页，打开的时候会自动调用这个`html`
  - 一般来说只需要复制黏贴即可，具体的后面会说
- `$name.vue`：放在`frontend/src/views/$name`文件夹下
  - 一般包含三个部分
    - template：组件的组织，相当于排版
    - script：引入模板中使用的组件（import后写到components里/可以参考已有文件的写法），定义一些需要用到的方法
    - style：就是一些个css写在这里
  - 页面的主要结构和方法都在这里面写，基本就拿`src/components`里面的组件拼一拼，或者没有的自己写一写就行了
  - 如果用到了ncr的组件的话，请务必在最后写上这一段

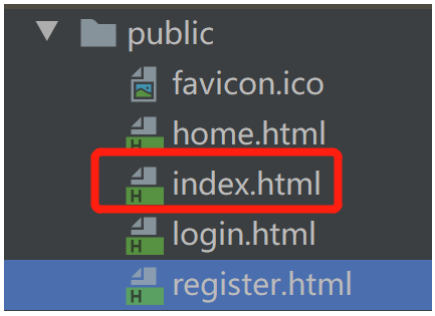
```
<style lang="scss">
    // Allow element/type selectors, because this is global CSS.
    // stylelint-disable selector-max-type, selector-class-pattern

    // Design variables and utilities from src/design.
    @import '../../design';
</style>
```

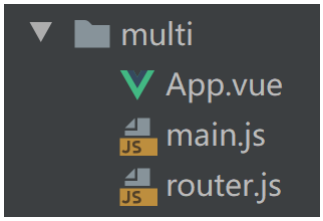
- `main.js`：放在`frontend/src/views/$name`文件夹下
  - 注册`vue`元素，将它渲染成一个组件
  - 一般来说只需要复制黏贴即可，具体的后面会说

## 有内部跳转的单页面

有内部跳转的页面的基本结构如下：（`multi`这个就是我拿来给大家当模板的，实际上没有在`vue.config.js`文件中设置，但是还是可以用后面的方法进行测试）



文件结构示例1



文件结构示例2

一共是四个文件：

- `$name.html`：同上，不做赘述
- `App.vue`：放在`frontend/src/views/$name`文件夹下
  - 与前面的`vue`文件有所不同的是，在这个`vue`文件中需要import的组件应该是这个页面中需要跳转的子页面，建议是子页面都写成components放到`src/components`文件夹中（我这里为了给出一个示例，所以调用了其他单页面）
  - 同样是包含三个部分，但内容会稍微有些不同
    - template: 这里不需要进行排版工作了，因为子页面都已经做好排版了，一般只需要一个`routerview`就可以了

      ```
      <template>
          <router-view></router-view>
      </template>
      ```

    - script：同样也不需要做方法的定义，这里只需要引入子页面并且写到`components`里面，同时还可以定义一些子页面需要共享的参数，子页面可以通过`$parent.paramsname`来访问
    - style：同上不做赘述
- `main.js`：放在`frontend/src/views/$name`文件夹下
  - 这里也是复制黏贴即可
- `router.js`：用来设置路由
  - 每一个子界面都需要在`router.js`中进行注册（需要引入哦）

    ```
    const routes = [
      {
        path: '/', //这个是第一个显示的页面
        name: 'first',
        component: first
      },
      {
        path: '/Login', //需要进入这个页面的话就用 this.$router.push('/Login')
        name: 'Login',
        component: Login
      },
    ```

```
  {
    path:'/Register',
    name: 'Register',
    component: Register
  }
];
```

- 我尝试了一下，我觉得这个路由是没有办法在不修改当前`url`的情况下进行跳转的，所以会出现他们说的刷新问题，如果非得要用这种有内部跳转的界面，可能需要尝试一下使用`onbeforeunload`这种操作，我不是很确定能不能用，可以去查一下（后来看了一下是不是用了history mode+后台覆盖的方法来解决，这个要不跟后端商量一下？）

# 运行机制

为了能让大家更快记住这些配置过程，我们先来了解一下运行机制。

每个界面的入口都是一个`main.js`文件，通过这个文件我们把所需的界面(`vue`元素)注册好了，然后给他们挂载一个名字，这里我拿有内部跳转的单页面的文件来举例，代码短一点看的清楚一点，有不同的地方我会注释。

```
import Vue from "vue";
import App from "./App"; //引入了`$name.vue`文件
import router from "./router"; //没有内部跳转的就把这个删掉
import axios from 'axios';
import BootstrapVue from 'bootstrap-vue';


Vue.use(BootstrapVue)
Vue.config.productionTip = false
Vue.prototype.$ajax = axios;
axios.defaults.baseURL = "http://127.0.0.1:8000/";

Vue.config.productionTip = false;

new Vue({
    router,// 没有内部跳转的就把这个删掉
    render: h => h(App) //这里就是新建了一个页面的元素
}).$mount("#multi"); // 给它挂载的名字是multi
```

然后我们就会通过`$name.html`文件来调用这个元素。

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,initial-scale=1.0">
<!--    <link rel="icon" href="<%= BASE_URL %>favicon.ico"> -->
    <title><%= htmlWebpackPlugin.options.title %></title>
  </head>
  <body>
    <noscript>
      <strong>We're sorry but <%= htmlWebpackPlugin.options.title %> doesn't
work properly without JavaScript enabled. Please enable it to continue.</strong>
    </noscript>
```

```
    <div id="multi"></div> //唯一需要改的地方就在这里 这里的id改成挂载的名字
    <!-- built files will be auto injected -->
  </body>

</html>
```

接下来我们再来看看我们的页面`$name.vue`文件。

```
<template>
    <router-view></router-view> //有跳转就这一句就够了 没有的话在这里对组件进行排版
</template>

<script>
    import first from "../first/first"; //引入组件，有跳转的话就是子页面，没有的话就是要用的组件
    import Login from "../login/Login";
    import Register from "../register/Register";

    export default {
        name: "App.vue", //这个地方的名字 就是你在main.js中引入的时候用的名字
        components: {
            first, //要用的组件一定要注册
            Login,
            Register
        }
        // 如果是有需要各个组件共享的参数，可以用
        // data: {
        //      return {
        //          这里写参数
        //      }
        // }
    }
</script>

<style scoped>
// 自己定义的css 如果用了ncr的组件一定要记得加那一段
</style>
```

# 环境配置

每次新建一个界面都要进行一次配置，这样才能生成对应的html文件，然后才能调用。

## `frontend/vue.config.js`

主要需要修改的就是下面这个地方

```
pages: {
        index: {
            entry: 'src/views/first/main.js',
            template: 'public/index.html',
            filename: 'index.html',
            chunks: ['chunk-vendors', 'chunk-common', 'index']
        },
        login: {
```

```
            entry: 'src/views/login/main.js',
            template: 'public/login.html',
            filename: 'login.html',
            chunks: ['chunk-vendors', 'chunk-common', 'login']
        },
        register: {
            entry: 'src/views/register/main.js',
            template: 'public/register.html',
            filename: 'register.html',
            chunks: ['chunk-vendors', 'chunk-common', 'register']
        },
        home: {
            entry: 'src/views/home/main.js',
            template: 'public/home.html',
            filename: 'home.html',
            chunks: ['chunk-vendors', 'chunk-common', 'home']
        },
    },
```

我这边随便拿一个来讲一下：

```
index: { //每个页面都需要一个这样的结构
            entry: 'src/views/multi/main.js', //放对应文件夹里的main.js
            template: 'public/index.html', //放public文件夹里对应的html文件
            filename: 'index.html',  //最后在dist文件夹内生成的东西
            chunks: ['chunk-vendors', 'chunk-common', 'index'] //这个不知道干嘛用的
        },
```

# 测试

一般来说大家自己先测试完了自己的界面之后再做整合会比较快一点，但是建议大家整合后再做一次测试。

测试的话因为现在没有写跟后端的交互，所以就测试一下是不是可以接上。只需要在index页面这里改几个地方就行。

```
index: {
            entry: 'src/views/multi/main.js', //这里的文件夹名改成你那个文件夹名就可以
            template: 'public/index.html',
            filename: 'index.html',
            chunks: ['chunk-vendors', 'chunk-common', 'index']
        },
```

每次修改了`frontend/vue.config.js`之后必须要重新`npm run serve`之后才能看到，不像`vue`文件可以直接重新自动渲染。