构建实战篇3：多页路由与模板解析

上篇文章中我们成功打包并输出了多页文件，而构建一个多页应用能够让我们进一步了解项目配置的可拓展性，可以对学习Vue 和webpack 起到强化训练的效果，本文将在此基础上主要针对多页路由及模板的配置进行系列的介绍。

路由配置

1. 跳转

在配置路由前，首先我们要明确一点就是，多页应用中的每个单页都是相互隔离的，即如果你想从page1下的路由跳到page2下的路由，你无法使用vue-router中的方法进行跳转，需要使用原生方法： `location.href` 或 `location.replace` 。

此外为了能够清晰的分辨路由属于哪个单页，我们应该给每个单页路由添加前缀，比如：

- index 单页：/vue/
- page1 单页：/vue/page1/
- page2 单页：/vue/page2/

其中/vue/ 为项目的二级目录，其后的目录代表路由属于哪个单页。因此我们每个单页的路由配置可以像这样：

javascript

```
/* page1 单页路由配置 */

import Vue from 'vue'
import Router from 'vue-router'

// 首页
const Home = (resolve => {
    require.ensure(['../views/home.vue'], () => {
        resolve(require('../views/home.vue'))
    })
```

```
let base = `${process.env.BASE_URL}` + 'page1'; // 添加单页前缀

export default new Router({
    mode: 'history',
    base: base,
    routes: [
        {
            path: '/',
            name: 'home',
            component: Home
        },
    ]
})
```

我们通过设置路由的base 值来为每个单页添加路由前缀，如果是index 单页我们无需拼接路由前缀，直接跳转至二级目录即可。

那么在单页间跳转的地方，我们可以这样写：

```html
<template>
  <div id="app">
    <div id="nav">
      <a @click="goFn('')">Index</a> |
      <a @click="goFn('page1')">Page1</a> |
      <a @click="goFn('page2')">Page2</a> |
    </div>
    <router-view/>
  </div>
</template>

<script>
export default {
    methods: {
        goFn(name) {
            location.href = `${process.env.BASE_URL}` + name
        }
    }
}
</script>
```

```javascript
this.$openRouter({
    name: name, // 跳转地址
    query: {
        text: 'hello' // 可以进行参数传递
    },
})
```

使用上述 `$openRouter` 方法我们还需要一个前提条件，便是将其绑定到Vue的原型链上，我们在所有单页的入口文件中添加：

```javascript
import { Navigator } from '../../common' // 引入 Navigator

Vue.prototype.$openRouter = Navigator.openRouter; // 添加至 Vue 原型链
```

至此我们已经能够成功模仿vue-router 进行单页间的跳转，但是需要注意的是因为其本质使用的是location 跳转，所以必然会产生浏览器的刷新与重载。

2. 重定向

当我们完成上述路由跳转的功能后，可以在本地服务器上来进行一下测试，你会发现Index 首页可以正常打开，但是跳转Page1、Page2 却仍然处于Index 父组件下，这是因为浏览器认为你所要跳转的页面还是在Index 根路由下，同时又没有匹配到Index 单页中对应的路由。这时候我们服务器需要做一次重定向，将下方路由指向对应的html 文件即可：

```
/vue/page1 -> /vue/page1.html
/vue/page2 -> /vue/page2.html
```

在vue.config.js中，我们需要对devServer进行配置，添加 `historyApiFallback` 配置项，该配置项主要用于解决HTML5 History API产生的问题，比如其rewrites选项用于重写路由：

```javascript
/* vue.config.js */

let baseUrl = '/vue/';
```

```
    devServer: {
        historyApiFallback: {
            rewrites: [
                { from: new RegExp(baseUrl + 'page1'), to: baseUrl + 'page1.html' },
                { from: new RegExp(baseUrl + 'page2'), to: baseUrl + 'page2.html' },
            ]
        }
    }

    ...
}
```

上方我们通过rewrites匹配正则表达式的方式将 `/vue/page1` 这样的路由替换为访问服务器下正确html文件的形式，如此不同单页间便可以进行正确跳转和访问了。最后需要注意的是如果你的应用发布到正式服务器上，你同样需要让服务器或者中间层作出合理解析，参考：HTML5 History模式#后端配置例子

而更多关于historyApiFallback的信息可以访问：connect-history-api-fallback

模板配置

上篇文章我们已经介绍了关于多模板的读取和配置，在配置html-webpack-plugin 的时候我们提到了自定义配置，这里我将结合模板渲染的功能来进行统一介绍。

1. 模板渲染

这里所说的模板渲染是在我们的html模板文件中使用html-webpack-plugin提供的default template语法进行模板编写，比如：

```html
<!DOCTYPE html>
<html>
  <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">
    <title>模板</title>
    <% for (var chunk in htmlWebpackPlugin.files.css) { %>
```

```
    <% } %>
  </head>
  <body>
    <div id="app"></div>
    <!-- built files will be auto injected -->

    <% for (var chunk in htmlWebpackPlugin.files.js) { %>
        <% if(htmlWebpackPlugin.files.js[chunk]) {%>
            <script type="text/javascript" src="<%= htmlWebpackPlugin.files.js[chunk
        <%}%>
    <% } %>
  </body>
</html>
```

以上我们使用模板语法手动获取并遍历htmlWebpackPlugin打包后的文件并生成到模板中，其中的 `htmlWebpackPlugin` 变量是模板提供的可访问变量，其有以下特定数据：

```json
"htmlWebpackPlugin": {
    "files": {
        "css": [ "main.css" ],
        "js": [ "assets/head_bundle.js", "assets/main_bundle.js"],
        "chunks": {
            "head": {
                "entry": "assets/head_bundle.js",
                "css": [ "main.css" ]
            },
            "main": {
                "entry": "assets/main_bundle.js",
                "css": []
            },
        }
    }
}
```

我们通过 `htmlWebpackPlugin.files` 可以获取打包输出的js及css文件路径，包括入口文件路径等。

需要注意的是如果你在模板中编写了插入对应js及css的语法，你需要设置 `inject` 的值为false来关闭资源的自动注入：

```
let conf = {
    entry: filePath, // page 的入口
    template: filePath, // 模板路径
    filename: filename + '.html', // 生成 html 的文件名
    chunks: ['manifest', 'vendor',  filename],
    inject: false, // 关闭资源自动注入
}

...
```

否则在页面会引入两次资源，如下图所示：

2. 自定义配置

一样的需求情况的话，我们使用自定义配置会比较方便。比如我们需要在生产环境模板中引入第三方统计脚本：

```javascript
/* vue.config.js */

module.exports = {
    ...

    pages: utils.setPages({
        addScript() {
            if (process.env.NODE_ENV === 'production') {
                return `
                    <script src="https://s95.cnzz.com/z_stat.php?id=xxx&web_id=xxx"
                `
            }

            return ''
        }
    }),

    ...
}
```

然后在页面模板中通过 `htmlWebpackPlugin.options` 获取自定义配置对象并进行输出：

```html
<% if(htmlWebpackPlugin.options.addScript){ %>
    <%= htmlWebpackPlugin.options.addScript() %>
<%}%>
```

同时你也可以针对个别模板进行配置，比如我想只在Index 单页中添加统计脚本，在Page1 单页中添加其他脚本，那么你可以给addScript 传入标识符来进行判断输出，比如：

```html
<% if(htmlWebpackPlugin.options.addScript){ %>
    <%= htmlWebpackPlugin.options.addScript('index') %>
<%}%>
```

同时为addScript 方法添加参数from：

```
            let url = "https://xxx";

            if (from === 'index') {
                url = "https://s95.cnzz.com/z_stat.php?id=xxx&web_id=xxx";
            }

            return `
                <script src=${url} language="JavaScript"></script>
            `
        }

        return ''
    }
```

这样我们就完成了自定义配置中的模板渲染功能。当然根据实际项目需求你的自定义配置项可能会更加复杂和灵活。

结语

通过2 小节的学习，相信大家对Vue 多页应用的构建已经有所了解。本文在第1 节的基础上重点介绍了多页路由及模板的配置，阐述了其与单页应用的不同之处，同时针对模板自定义配置的使用场景给出了简单的实例，希望大家在了解的基础上将下方的实例代码作为参考，进行相应的实战。

本案例代码地址：multi-page-project

思考& 作业

- 多页应用中各自的 `Vuex Store` 信息能实现共享吗?
- html-webpack-plugin 如何解析非.html 的模板，比如.hbs，应该如何配置?

留言

评论将在后台进行审核，审核通过后对所有人可见

冰凉冰凉

你好，我想问问，克隆了本案例代码。然后yarn run build，也修改了let baseUrl = '/dist/';为啥把他放到服务器上部署，能加载css和js，还是会空白。

▲ 0　　评论　2天前

666啊

每个page下面都要新建一个html 模板页吗?

▲ 0　　评论　9天前

丈母娘

请问一下作者，如果我要配置cdn该怎么操作，弄了半天没搞成功。感谢

▲ 0　　收起评论　18天前

> 劳卜　前端工程师@ TC
> 配置cdn？引入的话直接html中引入资源就行了
> 10天前
>
> 评论审核通过后显示　　评论

劳卜 前端工程师@ TC

主要区别在于前者可以控制模版的渲染规则

▲ 0　　评论　2月前

不凌卟乱。

请问下,你使用模板语法自己向html注入js和css,跟不写这样语法,自动注入的css,js文件,这2种方法有何区别?

▲ 0　　收起评论　2月前

> 劳卜　前端工程师@ TC

评论审核通过后显示 评论

Si

使用模版后页面一直加载，空白；另外，作者github 上的项目打开也是空白；

▲ 1 收起评论 3月前

劳卜 前端工程师@ TC

使用yarn run serve 进行预览，build 需要配置baseUrl 为正确生产环境路径

3月前

秋石君

回复 劳卜： 配置baseUrl为正确生产环境路径.是什么意思?

2月前

fn就是我

也卡在这里了，作者能讲解下不

2月前

评论审核通过后显示 评论

Si

多页面中单个页面的state 为啥无法保留?一刷新又回到初始值

▲ 0 收起评论 3月前

劳卜 前端工程师@ TC

因为数据是保存在运行内存中的，页面刷新会被重新赋值，可以参考：
https://blog.csdn.net/guzhao593/article/details/81435342

3月前

Si

谢谢!

评论审核通过后显示 评论

Si

这节的整个逻辑有点乱,希望能把每个文件的文件名加上.

▲ 1 收起评论 3月前

Si

删除此条评论

3月前

LYSEKY

不能统一都用index.html和index.js嘛，改文件夹就行，感觉需要改动的地方太多

2月前

评论审核通过后显示 评论