Vue.js 自定义指令 Vue.js 起步

Vue.js 路由

本章节我们将为大家介绍 Vue.js 路由。

Vue.js 路由允许我们通过不同的 URL 访问不同的内容。

通过 Vue.js 可以实现多视图的单页Web应用（single page web application，SPA）。

Vue.js 路由需要载入 vue-router 库

中文文档地址：vue-router文档。

安装

1、直接下载 / CDN

```
https://unpkg.com/vue-router/dist/vue-router.js
```

NPM

推荐使用淘宝镜像：

```
cnpm install vue-router
```

简单实例

Vue.js + vue-router 可以很简单的实现单页应用。

<router-link> 是一个组件，该组件用于设置一个导航链接，切换不同 HTML 内容。 **to** 属性为目标地址， 即要显示的内容。

以下实例中我们将 vue-router 加进来，然后配置组件和路由映射，再告诉 vue-router 在哪里渲染它们。代码如下所示：

HTML 代码

```
<script src="https://unpkg.com/vue/dist/vue.js"></script>
<script src="https://unpkg.com/vue-router/dist/vue-router.js"></script>
<div id="app">
<h1>Hello App!</h1>
<p>
<!-- 使用 router-link 组件来导航. -->
<!-- 通过传入 `to` 属性指定链接. -->
<!-- <router-link> 默认会被渲染成一个 `<a>` 标签 -->
<router-link to="/foo">Go to Foo</router-link>
<router-link to="/bar">Go to Bar</router-link>
</p>
<!-- 路由出口 -->
<!-- 路由匹配到的组件将渲染在这里 -->
<router-view></router-view>
</div>
```

JavaScript 代码

```
// 0. 如果使用模块化机制编程，导入 Vue 和 VueRouter，要调用 Vue.use(VueRouter)
// 1. 定义（路由）组件。
// 可以从其他文件 import 进来
const Foo = { template: '<div>foo</div>' }
const Bar = { template: '<div>bar</div>' }
// 2. 定义路由
// 每个路由应该映射一个组件。 其中"component" 可以是
// 通过 Vue.extend() 创建的组件构造器，
// 或者，只是一个组件配置对象。
// 我们晚点再讨论嵌套路由。
const routes = [
{ path: '/foo', component: Foo },
{ path: '/bar', component: Bar }
]
// 3. 创建 router 实例，然后传 `routes` 配置
// 你还可以传别的配置参数，不过先这么简单着吧。
const router = new VueRouter({
routes //  （缩写）相当于 routes: routes
})
// 4. 创建和挂载根实例。
// 记得要通过 router 配置参数注入路由，
// 从而让整个应用都有路由功能
const app = new Vue({
router
}).$mount('#app')
// 现在，应用已经启动了!
```

尝试一下 »

点击过的导航链接都会加上样式 `class ="router-link-exact-active router-link-active"`。

<router-link> 相关属性

接下来我们可以了解下更多关于 <router-link> 的属性。

to

表示目标路由的链接。 当被点击后，内部会立刻把 to 的值传到 router.push()，所以这个值可以是一个字符串或者是描述目标位置的对象。

```
<!-- 字符串 -->
<router-link to="home">Home</router-link>
<!-- 渲染结果 -->
<a href="home">Home</a>

<!-- 使用 v-bind 的 JS 表达式 -->
<router-link v-bind:to="'home'">Home</router-link>

<!-- 不写 v-bind 也可以，就像绑定别的属性一样 -->
```

```
<router-link :to="'home'">Home</router-link>

<!-- 同上 -->
<router-link :to="{ path: 'home' }">Home</router-link>

<!-- 命名的路由 -->
<router-link :to="{ name: 'user', params: { userId: 123 }}">User</router-link>

<!-- 带查询参数，下面的结果为 /register?plan=private -->
<router-link :to="{ path: 'register', query: { plan: 'private' }}">Register</router-link>
```

replace

设置 replace 属性的话，当点击时，会调用 router.replace() 而不是 router.push()，导航后不会留下 history 记录。

```
<router-link :to="{ path: '/abc'}" replace></router-link>
```

append

设置 append 属性后，则在当前 (相对) 路径前添加基路径。例如，我们从 /a 导航到一个相对路径 b，如果没有配置 append，则路径为 /b，如果配了，则为 /a/b

```
<router-link :to="{ path: 'relative/path'}" append></router-link>
```

tag

有时候想要 `<router-link>` 渲染成某种标签，例如 `<li>`。 于是我们使用 `tag` prop 类指定何种标签，同样它还是会监听点击，触发导航。

```
<router-link to="/foo" tag="li">foo</router-link>
<!-- 渲染结果 -->
<li>foo</li>
```

active-class

设置 链接激活时使用的 CSS 类名。可以通过以下代码来替代。

```
<style>
   ._active{
      background-color : red;
   }
</style>
<p>
   <router-link v-bind:to = "{ path: '/route1'}" active-class = "_active">Router Link 1</router-link>
   <router-link v-bind:to = "{ path: '/route2'}" tag = "span">Router Link 2</router-link>
</p>
```

注意这里 **class** 使用 `active_class="_active"`。

exact-active-class

配置当链接被精确匹配的时候应该激活的 class。可以通过以下代码来替代。

```
<p>
   <router-link v-bind:to = "{ path: '/route1'}" exact-active-class = "_active">Router Link 1</router-li
nk>
   <router-link v-bind:to = "{ path: '/route2'}" tag = "span">Router Link 2</router-link>
</p>
```

event

声明可以用来触发导航的事件。可以是一个字符串或是一个包含字符串的数组。

```
<router-link v-bind:to = "{ path: '/route1'}" event = "mouseover">Router Link 1</router-link>
```

以上代码设置了 event 为 mouseover ，及在鼠标移动到 Router Link 1 上时导航的 HTML 内容会发生改变。

NPM 路由实例

接下来我们演示了一个使用 npm 简单的路由实例，开始前，请先下载该实例源代码：

⬇ 路由实例

你也可以在 Github 上下载：https://github.com/chrisvfritz/vue-2.0-simple-routing-example

下载完后，解压该目录，重命名目录为 vue-demo，vu 并进入该目录，执行以下命令：

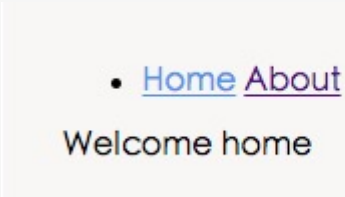
```
# 安装依赖，使用淘宝资源命令 cnpm
cnpm install

# 启动应用，地址为 localhost:8080
cnpm run dev
```

如果你需要发布到正式环境可以执行以下命令：

```
cnpm run build
```

执行成功后，访问 http://localhost:8080 即可看到如下界面：



Vue.js 自定义指令 Vue.js 起步

1 篇笔记

写笔记

exact-active-class 和 active-class 的区别

router-link 默认情况下的路由是模糊匹配，例如当前路径是 /article/1 那么也会激活 <router-link to="/article">，所以当设置 exact-active-class 以后，这个 router-link 只有在当前路由被全包含匹配时才会被激活 exact-active-class 中的 class，例如：

```
<router-link to="/article" active-class="router-active"></router-link>
```

当用户访问 /article/1 时会被激活为：

```
<a href="#/article" class="router-active" rel="nofollow"></a>
```

而当使用：

```
<router-link to="/article" exact-active-class="router-active"></router-link>
```

当用户访问 /article/1 时，不会激活这个 link 的 class：

```
<a href="#/article" rel="nofollow"></a>
```

lunatic 5个月前 (10-22)