# 7.路由vue-router

## 7.1.场景模拟

现在我们来实现这样一个功能：

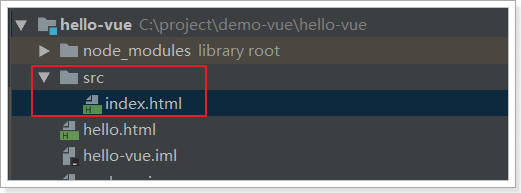
一个页面，包含登录和注册，点击不同按钮，实现登录和注册页切换：



### 7.1.1.编写父组件

为了让接下来的功能比较清晰，我们先新建一个文件夹：src

然后新建一个HTML文件，作为入口：index.html



然后编写页面的基本结构：

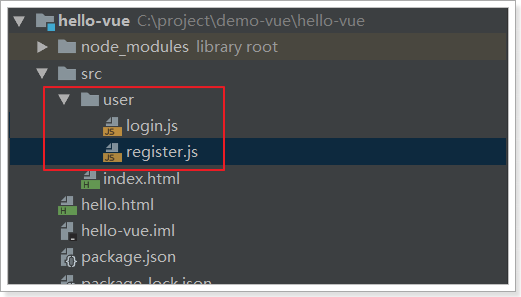
<div id="app">  
 <span>登录</span>  
 <span>注册</span>  
 <hr/>  
 <div>  
 登录页/注册页  
 </div>  
</div>  
<script src="../node\_modules/vue/dist/vue.js"></script>  
<script type="text/javascript">  
 var vm = new Vue({  
 el:"#app"  
 })  
</script>

样式：



### 7.1.2.编写登录及注册组件

接下来我们来实现登录组件，以前我们都是写在一个文件中，但是为了复用性，开发中都会把组件放入独立的JS文件中，我们新建一个user目录以及login.js及register.js：



编写组件，这里我们只写模板，不写功能。

login.js内容如下：

const loginForm = {  
 template:'\  
 <div>\  
 <h2>登录页</h2> \  
 用户名：<input type="text"><br/>\  
 密码：<input type="password"><br/>\  
 </div>\  
 '  
}

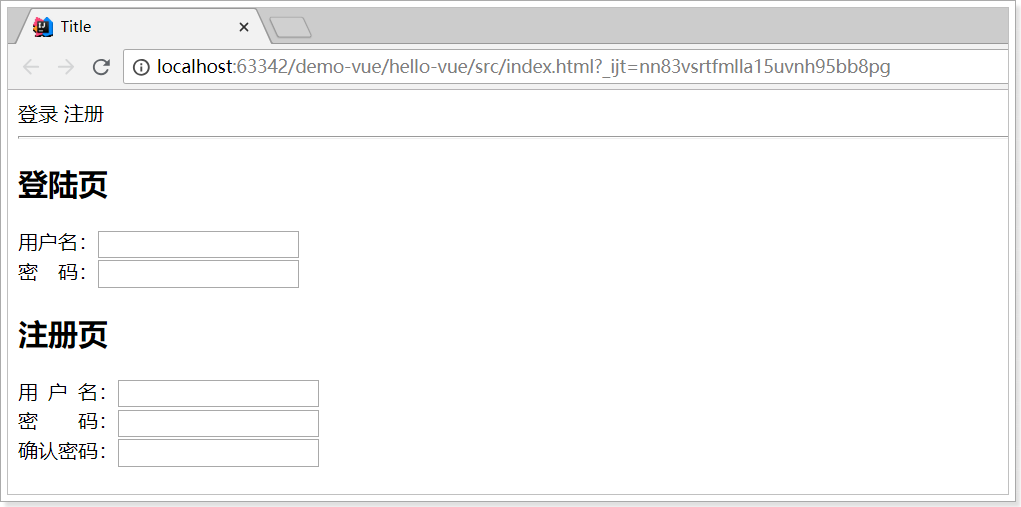
register.js内容：

const registerForm = {  
 template:'\  
 <div>\  
 <h2>注册页</h2> \  
 用&ensp;户&ensp;名：<input type="text"><br/>\  
 密&emsp;&emsp;码：<input type="password"><br/>\  
 确认密码：<input type="password"><br/>\  
 </div>\  
 '  
}

### 7.1.3.在父组件中引用

<div id="app">  
 <span>登录</span>  
 <span>注册</span>  
 <hr/>  
 <div>  
 <!--<loginForm></loginForm>-->  
 <!--  
 疑问：为什么不采用上面的写法？  
 由于html是大小写不敏感的，如果采用上面的写法，则被认为是<loginform></loginform>  
 所以，如果是驼峰形式的组件，需要把驼峰转化为“-”的形式  
 -->  
 <login-form></login-form>  
 <register-form></register-form>  
 </div>  
</div>  
<script src="../node\_modules/vue/dist/vue.js"></script>  
<script src="user/login.js"></script>  
<script src="user/register.js"></script>  
<script type="text/javascript">  
 var vm = new Vue({  
 el: "#app",  
 components: {  
 loginForm,  
 registerForm  
 }  
 })  
</script>

效果：



### 7.1.5.问题

我们期待的是，当点击登录或注册按钮，分别显示登录页或注册页，而不是一起显示。

但是，如何才能动态加载组件，实现组件切换呢？

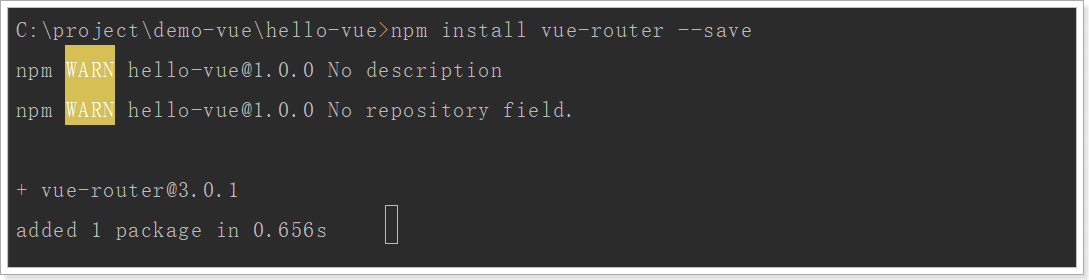
虽然使用原生的Html5和JS也能实现，但是官方推荐我们使用vue-router模块。

## 7.2.vue-router简介和安装

使用vue-router和vue可以非常方便的实现 复杂单页应用的动态路由功能。

官网：<https://router.vuejs.org/zh-cn/>

使用npm安装：npm install vue-router --save



在index.html中引入依赖：

<script src="../node\_modules/vue-router/dist/vue-router.js"></script>

## 7.3.快速入门

新建vue-router对象，并且指定路由规则：

// 创建VueRouter对象  
const router = new VueRouter({  
 routes:[ // 编写路由规则  
 {  
 path:"/login", // 请求路径，以“/”开头  
 component:loginForm // 组件名称  
 },  
 {  
 path:"/register",  
 component:registerForm  
 }  
 ]  
})

* 创建VueRouter对象，并指定路由参数
* routes：路由规则的数组，可以指定多个对象，每个对象是一条路由规则，包含以下属性：
  + path：路由的路径
  + component：组件名称

在父组件中引入router对象：

var vm = new Vue({  
 el:"#app",  
 components:{// 引用登录和注册组件  
 loginForm,  
 registerForm  
 },  
 router // 引用上面定义的router对象  
})

页面跳转控制：

<div id="app">  
 <!--router-link来指定跳转的路径-->  
 <span><router-link to="/login">登录</router-link></span>  
 <span><router-link to="/register">注册</router-link></span>  
 <hr/>  
 <div>  
 <!--vue-router的锚点-->  
 <router-view></router-view>  
 </div>  
</div>

* 通过<router-view>来指定一个锚点，当路由的路径匹配时，vue-router会自动把对应组件放到锚点位置进行渲染
* 通过<router-link>指定一个跳转链接，当点击时，会触发vue-router的路由功能，路径中的hash值会随之改变

效果：



**注意**：单页应用中，页面的切换并不是页面的跳转。仅仅是地址最后的hash值变化。

事实上，我们总共就一个HTML：index.html