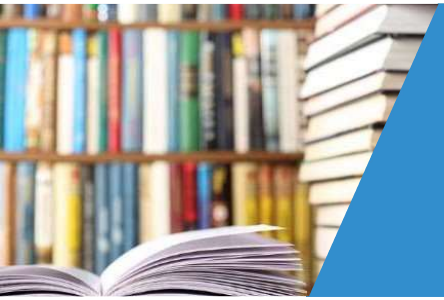
Vue-Router路由

软通大学
ISOFTSTONE UNIVERSITY

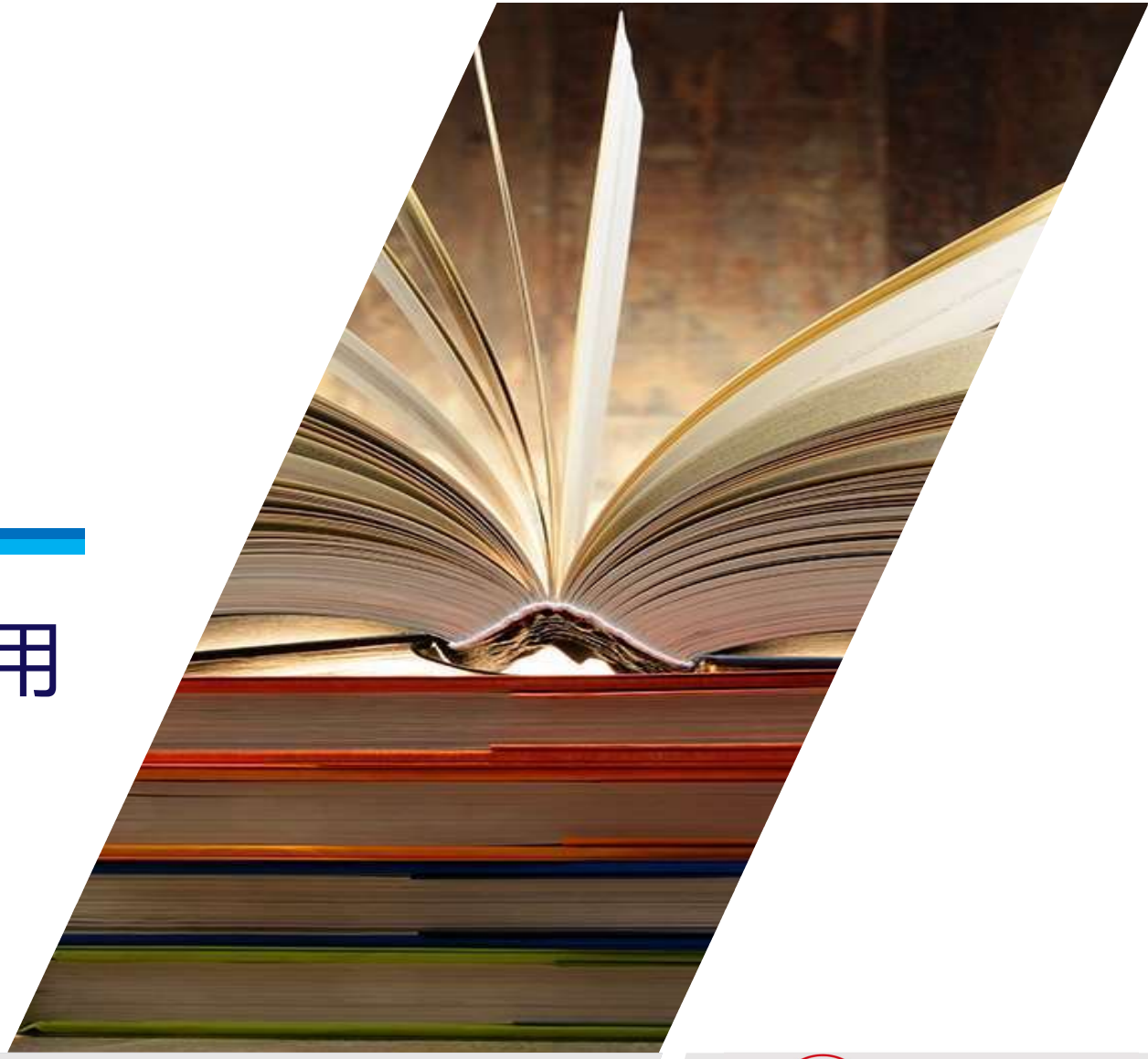
本节目标

- 理解前端路由
- Vue-Router的使用



目录 CONTENTS

01 Vue-Router基本使用



Vue-Router基本使用

前端路由的概念:

对于普通的网站，所有的超链接都是URL地址，所有的URL地址都对应服务器上对应的资源, 而对于单页面引用程序来讲，主要通过URL中的hash(#号)来实现不同页面之间的切换，同时，hash有一个特点：HTTP请求中不会包含hash相关的内容；所以，单页面程序中的页面跳转主要用hash实现。在单页面应用程序中，这种通过hash改变来切换页面的方式，称作前端路由, Vue-Router就是Vue实现SPA前端路由的js插件。

Vue-Router的基本使用:

1. 安装Vue-Router:
   cnpm i vue-router –D
2. 导入路由组件，并注册全局使用:
   import VueRouter from "vue-router“ Vue.use(VueRouter);
3. 导入路由需要的组件: 比如
   import LyLogin from "./components/LyLogin";
   import LyRegister from "./components/LyRegister“
4. 创建路由对象，定义路由规则
5. 将路由对象挂载到vm实例上

Vue-Router基本使用

~4:路由基本规则定义:

```
var routerObj = new VueRouter({
  //routes属性：表示配置路由匹配规则
  routes: [
    {
     //path， 表示监听 哪个路由链接地址
      path: "/login",
     //component， 表示 path对应要展示 的组件
      component: LyLogin
    },
    {
      path: "/register",
      component: LyRegister
    }
  ]
});
```

~5:将routerObj挂载到vm实例上

```
var vm = new Vue({
  el: "#app",
  render: c => c(App4),
  //将路由对象挂载到 vm 上
  router:routerObj
})
```

```
<template>
  <div>
    <!-- 定义前端路由链接 -->
    <router-link to="/login">登录</router-link>
    <router-link to="/register">注册</router-link>
    <!-- router-view 是一个占位符，是 vue-router 提供的元素，将来，路由规则，匹配到的组件，就会展示到这个 router-view 中去 -->
    <router-view></router-view>
  </div>
</template>
```

Vue-Router基本使用

路由模块化封装:

在入口js文件中，与路由相关的代码，我们可以封装到一个独立的router.js文件中，并对外暴露。

```
import VueRouter from "vue-router"
import LyLogin from "./components/LyLogin";
var routerObj = new VueRouter({
routes: [
   {
     path: "/login",
     component: LyLogin
   }]
});
//将路由对象暴露出去
export default routerObj;
```

(router.js)

```
//引入自己封装的router.js路由模块
import VueRouter from "vue-router"
Vue.use(VueRouter);
import routerObj from "./router.js"

var vm = new Vue({
 el: "#app",
 render: c => c(App4),
 //将路由对象挂载到 vm 上
 router:routerObj
})
```

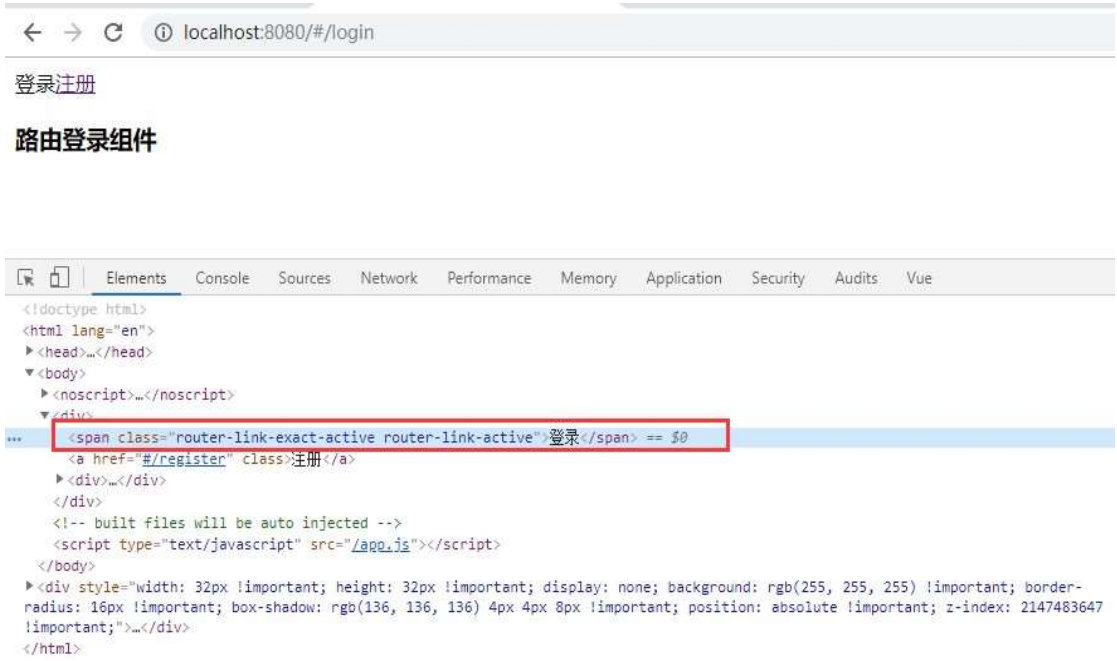
(main.js)

Vue-Router基本使用

tag属性指定route-link渲染标签:

默认router-link渲染的是a 标签，使用tag属性，可以指定渲染的标签元素:

```
<router-link to="/login" tag="span">登录</router-link>
```



Vue-Router基本使用

设置路由重定向:

默认router-view展示的是路由规则对应的组件内容，设置当项目启动根目录，展示登录组件:

```
var routerObj = new VueRouter({
routes: [
  {
    //为/路径设定重定向
    path:"/",redirect:"/login"
  },
  {
    path: "/login",
    component: LyLogin
  },
  {
    path: "/register",
    component: LyRegister
  }
 ]
});
```

Vue-Router基本使用

设置路由高亮:

为route-link当前点击选中的元素设置高亮效果：默认路由对象提供了linkActiveClass属性，对应router-link-active样式作为当前route-link选中样式, 有2种方式更改:

- 声明一个样式router-link-active覆盖默认样式
- linkActiveClass指定自定义的样式

```
<style>
.router-link-active{
  color: red;
  font-weight: 800;
  font-style: italic;
  font-size: 80px;
  text-decoration: underline;
}
</style>
```

(覆盖默认样式)

```
import democss from "./assets/demo.css"
import routerObj from "./router.js"
```

```
...
var routerObj = new VueRouter({
routes: [
  {path:"/xx",component:xx},{...}],
  linkActiveClass:'myactive'
});
```

(指定自定义样式)

*:demo.css中定义myactive样式,demo.css文件可以再main.js中导入，也可以再router.js中导入，只要routerObj路由对象可以找到就可以。

Vue-Router基本使用

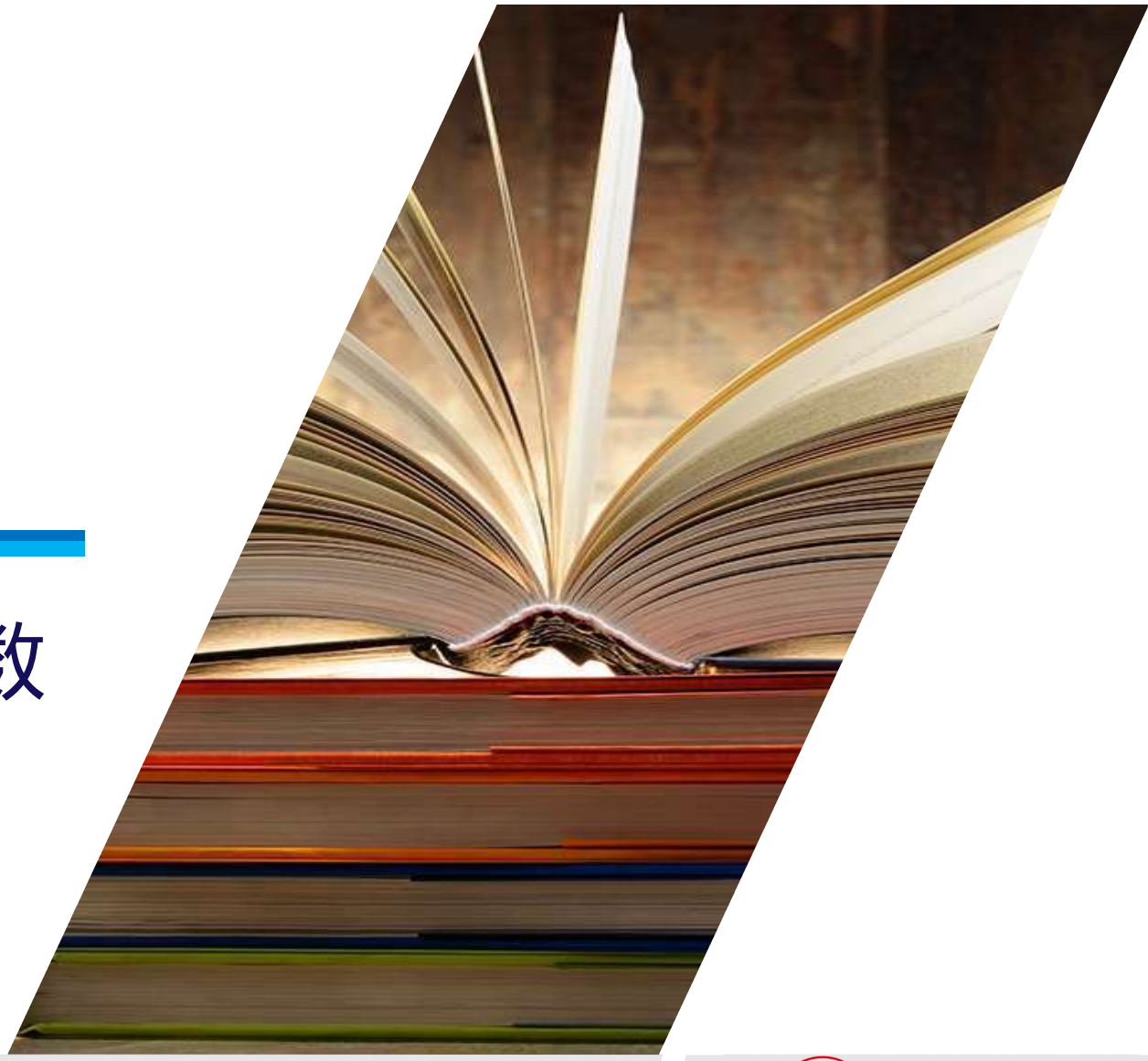
设置路由切换动画:

路由切换动画，和组件切换动画是一个方法，将代表组件的router-view标签用transition标签包裹，并提供过渡样式6个css

```
<template>
  <div>
      <router-link to="/login">登录</router-link>
      <router-link to="/register">注册</router-link>
      <transition mode="out-in">
       <router-view></router-view>
      </transition>
  </div>
</template>
```

```
<style>
.v-enter {opacity: 0;transform: translateX(150px);}
.v-enter-to {opacity: 1;transform: translateX(0px);}
.v-enter-active {transition: all 0.8s ease;}
.v-leave {opacity: 1;transform: translateX(0px);}
.v-leave-to {opacity: 0;transform: translateX(150px);}
.v-leave-active {transition: all 0.8s ease;}
/* v-move和v-leave-active绝对定位解决组件切换的错位问题 */
.v-move {transition: all 0.8s ease;}
.v-leave-active {position: absolute;}
</style>
```

02 路由规则中传递参数



路由规则中传递参数

路由规则中传递参数，支持? query传参 或者 restful风格的参数传递

? query查询字符串形式

在定义前端路由链接地址时，使用? a = xx & b= xx 的形式传递参数，不需要更改路由匹配规则

```
<router-link to="/login?id=1&name=zs">登录</router-link>
```

```
...
var router = new Vue.router({
routes:[{path:"/login",component:Login}]})
...
*: 路由规则中，不需要更改匹配规则
```

路由链接对应展示的Login.vue组件中获取链接参数：

```
<template>
 <div>
    <h3>路由登录组件</h3>
    <!-- 获取参数值 -->
    {{this.$route.query.id}}---{{this.$route.query.name}}
 </div>
</template>
```

路由规则中传递参数

restful风格的params方式传参，需要修改路由匹配规则，比如:

```
<router-link to="/register/admin/123456">注册</router-link>
```

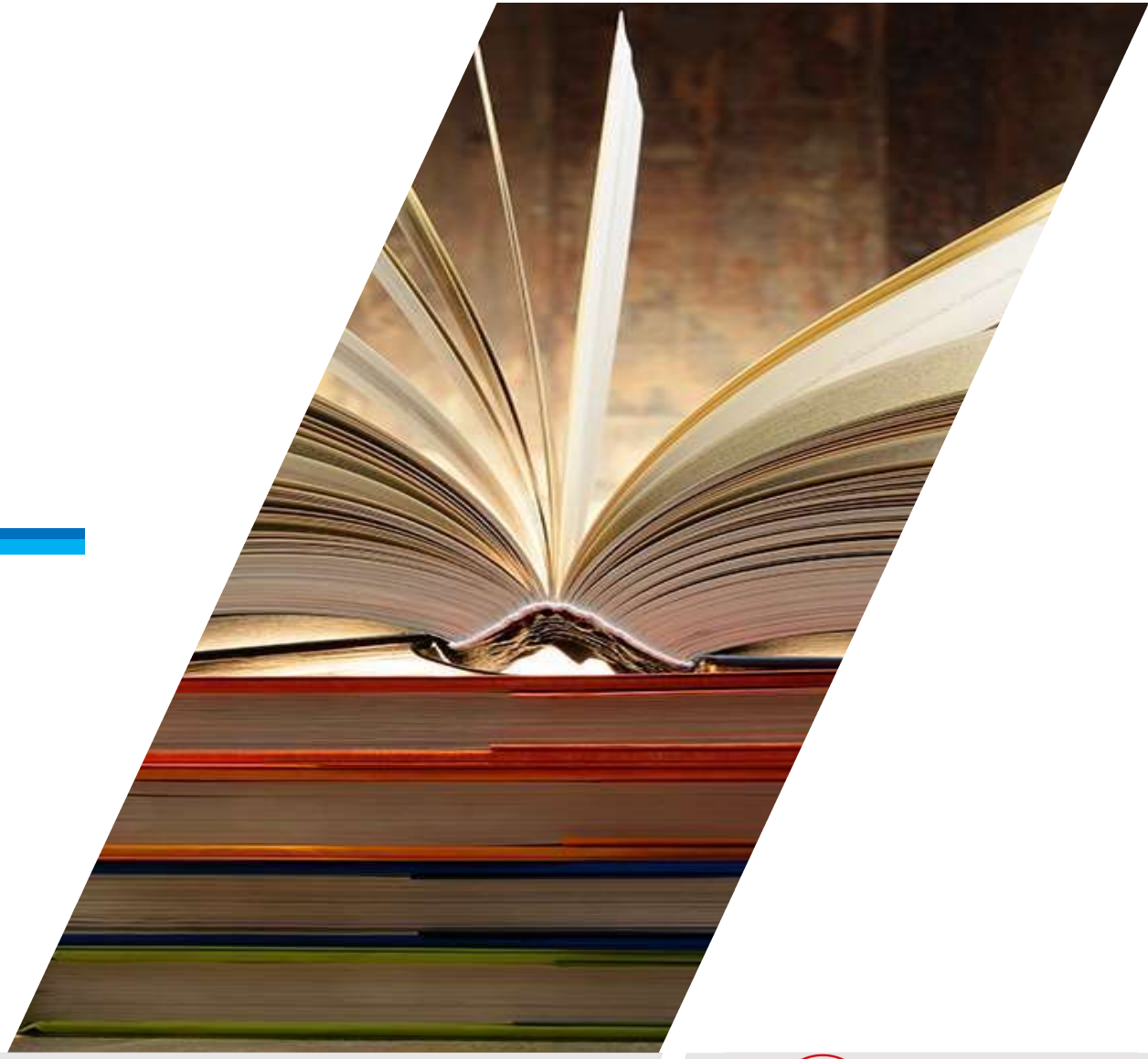
```
...
var routerObj = new VueRouter({
routes: [
  {
    ...
  },
  {
    path: "/register/:username/:password",
    component: LyRegister
  }
 ],
});
```

链接对应的register.vue组件内获取传递参数:

```
<template>
 <div>
   <h3>路由注册组件</h3>
   <!-- 获取params参数 -->
   {{this.$route.params.username}}---{{this.$route.params.password}}
 </div>
</template>
```

*: params方式传参数，routes路由规则需要改变

03 路由的嵌套



路由的嵌套

比如App5.vue默认展示的只有一个组件Accout.vue组件，而Accout.vue组件内包含了登录组件和注册组件，换言之，登录和注册组件的路由链接是属于Accunt父组件内部的。

App.vue组件

```
<div>
  <router-link to="/account">Account</router-link>
  <router-view></router-view>
</div>
```

Account组件

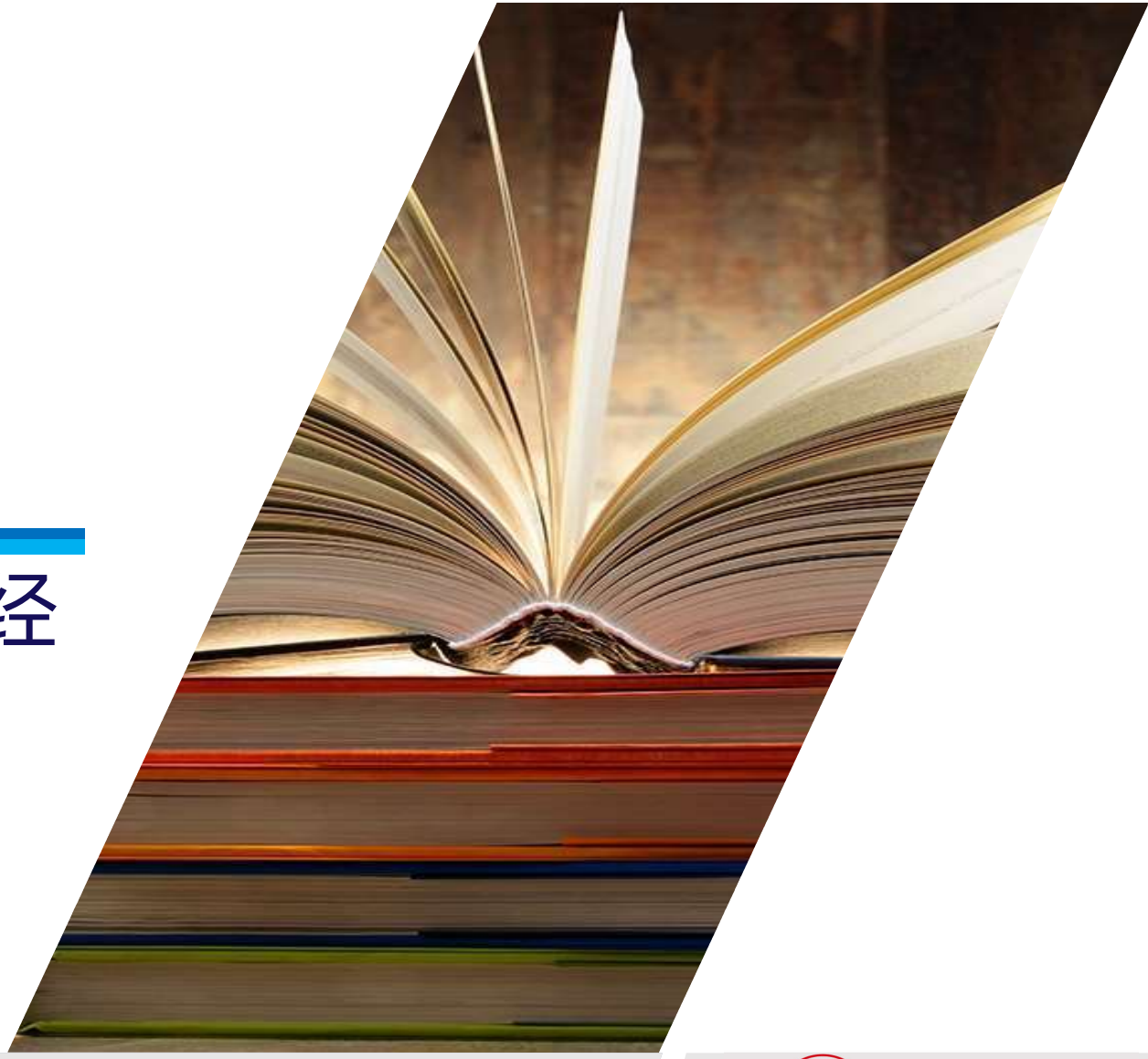
```
<div>
  <h3>我是Account组件</h3>
  <router-link to="/account/login?id=1&name=zb">登录</router-link>
  <router-link to="/account/register/admin/123456">注册</router-link>
  <router-view></router-view>
</div>
```

router.js路由匹配规则

```
var router = new VueRouter({
 routes: [
  {path: "/account",component: Account,
   children:[
    {path:"login",component:LyLogin},
    {path:"register/:username/:password",
     component:LyRegister}
   ]
  }
 ]
})
//将路由对象暴露出去
export default router;
```

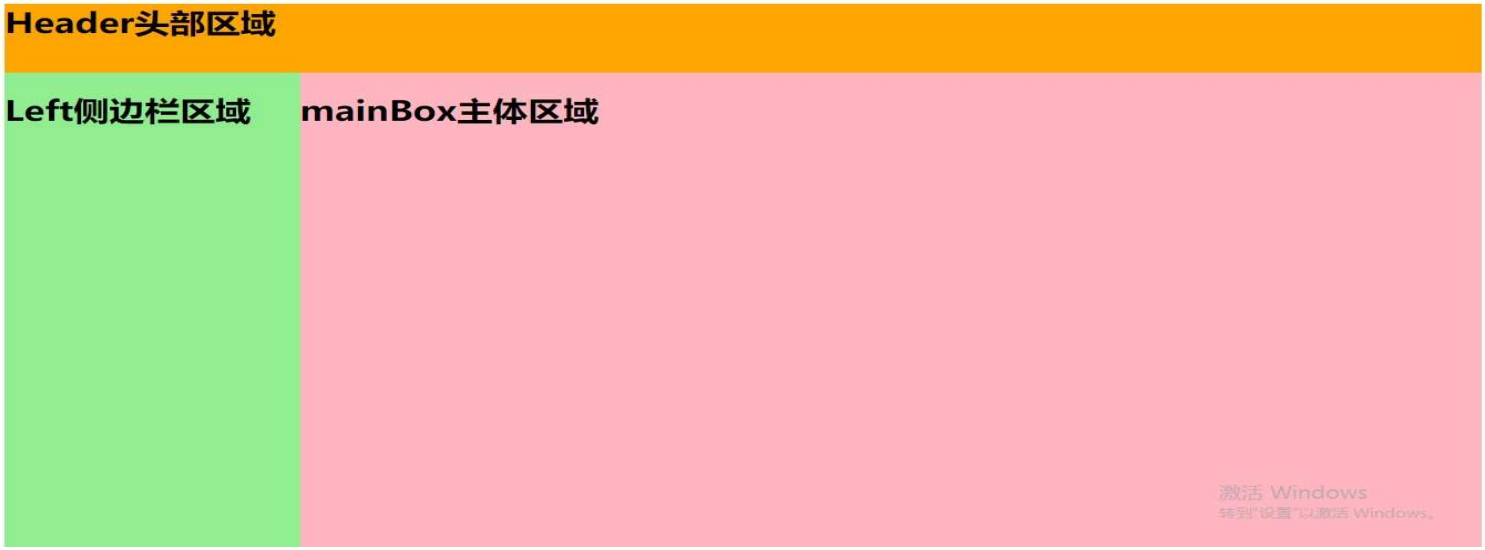
children 属性，实现子路由，同时，子路由的 path 前面，不要带 /，否则永远以根路径开始请求，这样不方便我们用户去理解URL地址

04 路由命名视图(实现经典布局)



路由命名视图

所谓命名视图，就是给router-view设置了name属性，路由规则中，当匹配到路由链接，展示对应组件到router-view时，通过components属性设定组件匹配规则，router-view会根据此规则展示组件，而不是默认全部占用。比如：要求实现如下样式的经典后台布局



*: Header、Left、mainBox是不同的三个组件，他们在入口组件中通过router-view命名视图占用。

路由命名视图

```
<div>
  <!-- 默认视图 -->
  <div class="header">
    <router-view></router-view>
  </div>
  <div class="container">
    <router-view name="left"></router-view>
    <router-view name="main"></router-view>
  </div>
</div>
```

(App.vue入口组件)

```
var router = new VueRouter({
 routes: [
  // 命令视图，components属性对应多组件配置
  {
    path: "/",
    components: {
      "default":Header,
      "left":Left,
      "main":Main
    }
  }
 ]
})
//将路由对象暴露出去
export default router;
```

*: 路由命名视图，path对应的组件属性不是component,而是components:{name:cname},配置name对应的cname组件

路由命名视图

路由命名视图的**style**样式:

```
<style lang="scss" scoped>
.header { background-color: orange; height: 80px; }

.container {
  display: flex;
  height: 600px;
  div:nth-child(1) {
    background-color: lightgreen;
    flex: 2;
  }
  div:nth-child(2) {
    background-color: lightpink;
    flex: 8;
  }
}
</style>
```

*: style的lang属性表示支持scss语法, 但是VueCli3.x项目默认是没有包含处理scss语法的依赖库的，所以需要导入处理scss语法的相关库.

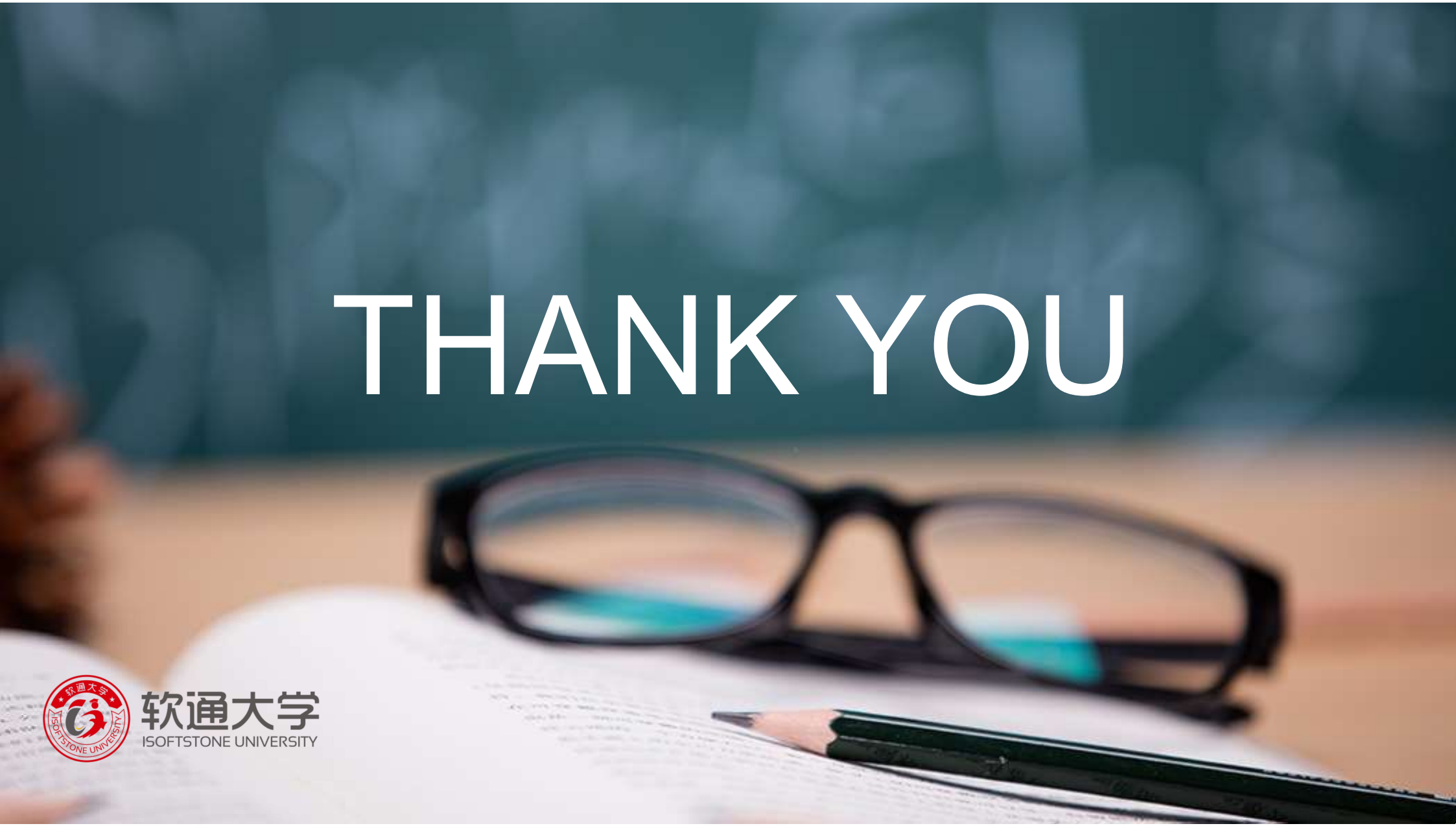
1: **cnpm install sass-loader**
2: cnpm install node-sass@latest

本节总结

- 前端路由的概念
- Vue-Router的基本使用：包含导入注册、路由规则定义、模块化封装、设置高亮、重定向、设置切换动画。
- 路由链接传参
- 子路由嵌套配置
- 路由命名视图

本节练习

- 课堂案例: Account组件内包含登录、注册组件，并分别传递参数到子组件，组件切换时加入动画，并设置高亮效果?



软通大学
ISOFTSTONE UNIVERSITY