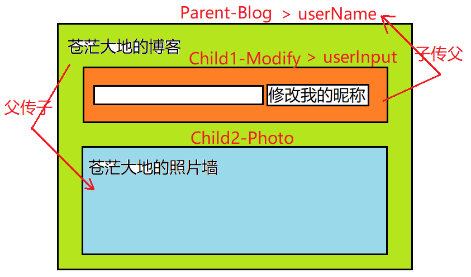
面试题：Angular模块中可以声明哪些种类的对象？

组件、指令、管道、服务(providers)、其它模块(imports)、fn、value

复习：

见思维导图

练习：实现如下的父子组件关系：



ng g component myc01-parent-blog

ng g component myc02-child1-modify

ng g component myc03-child2-photo

1.父子间组件传递——重点&难点

Vue.js和Angular中的父子间消息传递原理一样，都可以用口诀：

**“Props Down, Events Up”**

**方向1：父=>子**

父组件通过“子组件的自定义属性”向下传递数据给子组件

①Parent.ts: 父组件创建数据

userName='苍茫大地'

②Parent.html： 父组件将自己的数据绑定给子组件的属性

<child2 [child2Name]="userName"></child2>

-----------------------------------------------------------

③Child2.ts：子组件定义扩展属性

@Input( ) //声明为“输入型属性”

private child2Name = null

④Child2.html：子组件使用自定义属性

<p>{{child2Name}}<p>

**方向2：子=>父**

子组件通过触发特定的事件(其中携带着数据)，把数据传递给父组件(父组件提供事件处理方法)

①Child1.ts： 自定义事件发射器 —— 输出型属性

@Output( )

private cryEvent = new EventEmitter()

②Child1.ts： 在某个时刻发射事件

this.cryEvent.emit( 子组件想传给父组件的数据 )

---------------------------------------------------------------

③Parent.html：监听子组件的事件

<child1 (cryEvent)="doCry( $event )"></child1>

④Parent.ts：接收子组件传递的数据

doCry( e ){ e就是子组件想传给父组件的数据 }

|  |
| --- |
| **父子组件传递数据的简便方法：父亲直接使用子组件的引用——视图子组件**  Parent.html: 使用#为子组件声明识别符  <div #c1></div>  <myc02 #c2 ></myc02>  Parent.ts：  **@ViewChild( 'c1', {static:true} )**  private child1;  **@ViewChild( 'c2', {static:true} )**  private child2;  提示：ViewChild装饰器用于将子组件识别符与某个属性关联起来，第一个参数必须是已经存在的子组件识别符(不带#)，第二个参数static指定该组件是否为“静态组件”——不会有时有有时无的组件 |
| 注意：通过“ViewChild-视图子组件”方式父组件可以获得任一子组件中的任意数据——一定程度上违法了“最少知识法则” |

2.总结：组件的模板中可以出现的内容

**<div id="d">**

**<myc01 #c01></myc01>**

**<p \*ngIf="isMarried"></p>**

**<li \*ngFor="let emp of empList">{{emp}}</li>**

**<button (click)="doClick( $event )"></button>**

**<div #myDiv></div>**

**</div>**

========================================

①HTML标准元素/属性——如div/id...

②组件对应的元素——如myc01

③Angular指令——如ngIf、ngFor

④模型数据/方法——Model，如empList

⑤事件对象/子组件传递的数据——$event

⑥模板变量——在模板中声明的临时变量，如ngFor循环产生的emp、index等

⑦子元素识别符——都是以#开头，在脚本中使用ViewChild进行关联

3.路由和导航

**多页面应用：**一个项目中有多个完整HTML文件，使用超链接跳转——销毁一棵DOM树，同步请求另一棵，得到之后再重建新的DOM树；不足：DOM树要反复重建，间隔中客户端一片空白

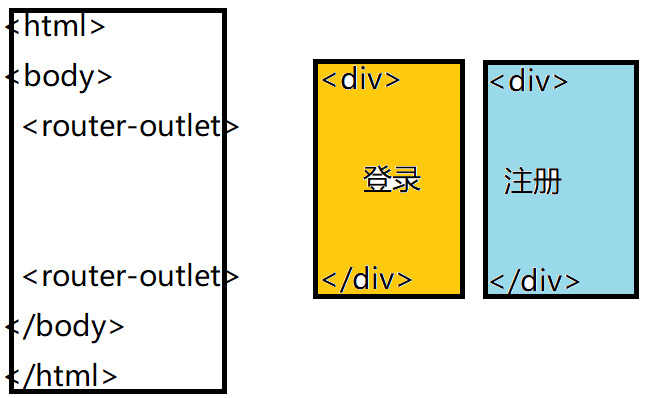


**单页面应用：**称为SPA(Single Page Application)，整个项目中有且只有一个“完整的”HTML文件，其它的“页面”都只是DIV片段；需要哪个“页面”就将其异步请求下来，“插入”到“完整的”HTML文件中



单页应用的优势：整个项目中客户端只需要下载一个HTML页面，创建一个完整的DOM树，页面跳转都是一个DIV替换另一个DIV而已——能够实现过场动画

单页应用的不足：不利于SEO访问优化



|  |
| --- |
| route：路径、路线、路由，有两部分：目标地址+处理过程  router：路由器，内部包含着路由词典 |

Angular中使用“单页应用”的步骤：

(0)创建整个应用需要的路由组件

ng g component index

ng g component product-list

ng g component product-detail

ng g component user-center

(1)定义“路由词典”——[{URL-组件}, {URL-组件}]

//app.module.ts 为每个路由组件分配一个路由地址

let routes = [

{ path: 'index', component: IndexComponent },

....

{ path: 'ucenter', component: UserCenterComponent },

]

(2)在根组件中注册“路由词典”

//app.module.ts

imports: [ BrowserModule, RouterModule.forRoot(routes)]

(3)创建路由组件挂载点——称为“路由出口”

//app.component.html

<router-outlet></router-outlet>

(4)访问测试

http://127.0.0.1:4200/ucenter

注意：

①路由词典中的路由地址不能以/开头或结尾，但中间可以包含/

②路由词典中可以指定一个默认首页地址： { path:"", component:...}

③路由词典中每个路由中要么指定 component(由哪个组件提供内容)，要么指定 redirectTo(重定向到另一个路由地址)

④路由词典中可以指定一个匹配任意地址的地址：'\*\*'，注意该地址只能用于整个路由词典的最后一个！！

|  |
| --- |
| 面试题：Vue.js中的路由跳转的机制有哪些？  **①hash法：只需要修改url中的hash部分**  http://127.0.0.1/index.html#/ucenter  **②history法：需要修改window.history对象，从而支持浏览器自带的后退按钮**  http://127.0.0.1/ucenter |
| 面试题：Angular中的路由跳转的机制有哪些？  **②history法：需要修改window.history对象，从而支持浏览器自带的后退按钮** |

4.路由跳转/导航

Navigate：导航，指从一个路由地址跳转到另一个路由地址

注意：不能直接使用<a href="/xx">方式跳转——会导致DOM树重建

可用的方案：

**方式1：使用模板跳转方法**

<any **routerLink**="/ucenter"></any>

注意：①可用于任意标签上 ②跳转地址应该以/开头，防止以相对方式跳转

**方式2：使用脚本跳转方法**

constructor(private router: Router){ } //声明依赖

jump(){

this.router.**navigateByUrl( '/ucenter' )** //执行跳转

}

注意：Router类是RouterModule提供的一个服务类，声明依赖即可使用

练习：在首页添加一个超链接，可以跳转到商品列表；在商品列表页面下创建三个超链接，表示三个商品，都可以跳转到商品详情页

5.路由参数

在路由词典中定义路由地址时，其中可以包含可变的参数：

{ path: 'product/detail/:lid' , component:....}

在路由跳转时可以为路由参数提供具体的参数值：

<a routerLink="/product/detail/5">

<a routerLink="/product/detail/13">

到了目标路由组件，可以读取“当前路由地址”中的路由参数：

constructor(private route: ActivatedRoute){ }

ngOnInit(){

//读取路由参数

this.route.params.subscribe( (data)=>{ data.lid就是路由参数 } )

}

课后任务：

(1)比较Vue.js和Angular中的路由系统的异同

(2)查看Angular阶段项目资料，创建出下列组件：

ng g component index 首页(1F/2F/3F)

ng g component product-list 商品列表页

ng g component product-detail 商品详情页

ng g component cart 购物车页

ng g component user-login 登录页

ng g component not-found 404页

(3)使用Angular路由系统，为上述组件分配合适的路由词典，并实现合理的路由跳转

(4)使用HttpClient调用服务器端API，获取出上述页面需要的数据