复习：

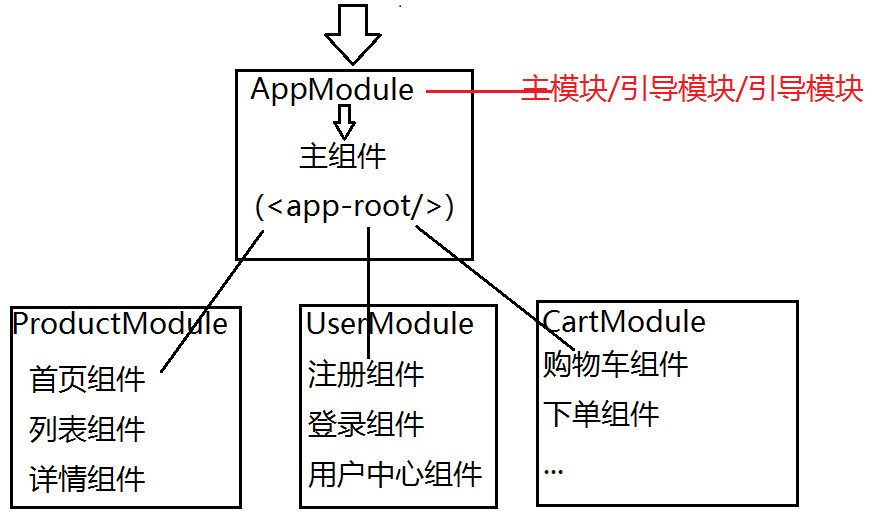
Angular是一个渐进式的MV\*框架，用于数据操作为主Web、移动端、桌面端应用。

|  |
| --- |
| Modal：模态对话框 window.alert()  Module： 模块  Model：模型，MV\*中的M |

1.面试题：Angular核心概念

**(1)模块：NgModule** —— Vue.js中无此概念

与ES和Node.js中的“模块”概念(一个文件就是模块，可以导出内部成员，引入其它模块成员)不同；NG中的模块指的是项目中的功能模块，如商品模块、用户模块等，其中包含着若干的组件、服务等对象



**(2)组件：Component**

组件是一块可重复使用的页面区域，有专有的样式、模板、数据等，表现上就是一个自定义的标签

(3)模板

(4)指令

(5)数据绑定

(6)服务和依赖注入

(7)路由



|  |
| --- |
| Vue中如何定义组件  Vue.component('组件名', {  template: '<div></div>',  components: { ... },  data(){return { } },  methods:{ doLogin(){} }  }) |
| <组件名 /> |

2.创建自定义组件

**(1)创建自定义组件**

//装饰器(Decorator)

@Component({ //元数据(描述数据的数据)对象

selector: 'my-login',

template: '<div>...</div>',

styleUrls: [ ]

})

export class Login{

uname = 'dingding'; //对象属性：Model数据

doLogin(){ } //对象方法：Model方法

}

**(2)注册自定义组件（在某个模块中，如AppModule：app.module.ts）**

@NgModule({

declarations: [ Login ]

})

**(3)使用自定义组件（自定义组件不能直接用于index.html，只能在主组件中使用，即AppComponent：app.component.html）**

<my-login></my-login>

练习：创建一个自定义组件：Main，在主组件中使用该组件

练习：创建一个自定义组件：ProductList，在Main组件中使用该组件，试着给它添加专有样式

3.Angular中的数据绑定

Angular中的数据绑定有四种情形：



(1)Model到View：插值表达式 {{ }}

(2)Model到View：属性绑定 [ ] 不再是 :

(3)View到Model：事件绑定 ( ) 不再是 @

(4)双向绑定：ngModel [(ngModel)]

注意：{{}}插值表达式中与Vue的不同点：1)不能使用new关键字，如new Date()；2)不能使用全局函数，如parseInt() 3)不能使用全局对象，如window/JSON

练习：自定义组件 ImageBox，其中实现如下效果：



提示：<img [src]="imgUrl"> 和 <button (click)="changeImg()">

练习：自定义组件 EmpInfo，使用 {{ }} 插值表达式把如下的数据绑定在视图中：

emp = {

ename: '丁然',

salary: 8000, 绑定出年薪

sex: 0, 绑定出男或女

birthday: 1501234567890 绑定出2017

projectList: ['达内OA', '联想ERP', '电信自动化结算'] 列表中显示

}

|  |
| --- |
| Vue.js提供了13个预定义指令  v-for、v-if、v-else-if、v-else、v-show  v-bind/:、v-on/@、v-html、v-text、v-cloak、v-once、v-pre  v-model |

4.Angular中的预定义指令 —— 小难点

Angular中的指令分为三类：

**(1)组件：** NG中组件继承自指令，是一种特殊的指令

Component extends Directive

**(2)结构型指令**：会影响模板最终的DOM树结构的指令，必须以\*开头

\*ngFor、\*ngIf、\*ngSwitchCase、\*ngSwitchDefault

**(3)属性型指令**：会影响当前DOM元素的属性的指令，必须用[ ]括起来

[ngClass]、[ngStyle]、[ngSwitch]、[(ngModel)]

**(1)ngFor：对数组或对象进行遍历**

<li \*ngFor="let n of myArr">{{n}}</li>

<li \*ngFor="let n of myArr; let i=index;">{{i}}:{{n}}</li>

练习：创建自定义组件EmpList，其中有属性/模型数据：

elist = [

{eid: 101, ename: 'dignding', sex: 1 }

{eid: 102, ename: 'dangdang', sex: 0 }

{eid: 103, ename: 'doudou', sex: 1 }

]，显示模板如下：



**(2)ngIf：赋值为true则挂载指定元素，false则删除指定元素**

<any \*ngIf="变量"></any>

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

<any \*ngIf="变量; else myelseblock"></any>

<ng-template #myelseblock >

<any>...</any>

</ng-template>

**(3)ngSwitch/ngSwitchCase/ngSwitchDefault： 实现多条件判定**

**注意：这三个指令的类型不一样**

|  |
| --- |
| JS: var status = 10;  switch( status ){  case 10:  ...  break;  case 20:  ...  break;  default:  ....  } |

<div [ngSwitch]="orderStatus">

<p \*ngSwitchCase="10">等待付款...</p>

<div \*ngSwitchCase="20">备货中...</div>

<span \*ngSwitchCase="30">运输中...</span>

<b \*ngSwitchCase="40">派送中...</b>

<i \*ngSwitchCase="50">已完成...</i>

<div \*ngSwitchDefault>不可识别的订单状态！</div>

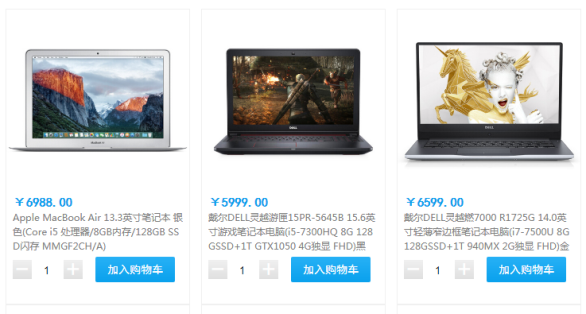
</div>

**(4)ngStyle和ngClass：可以绑定为String或Object**

课后任务：

(1)仿写官方示例项目“英雄指南”：https://angular.cn/tutorial

(2)完成学子商城前台系统“商品列表”功能点



提示：在自定义组件XzProductList，其中创建模型数据 productList: [

{ pid: 101, title:'商品名称', price:5999, count: 1},... ]；完成数据的绑定，并实现点击“+/-”购买数量发生改变。“加入购物车”按钮改为“删除此商品”，点击实现商品删除功能。