复习：

见思维导图

1.小知识：TS对ES的扩展

(1)强类型语言

(2)访问权限控制修饰符

(3)接口

(4)静态成员

**实例属性**：应该无具体值，通过对象应用访问，每个实例都有一份

**静态属性**：应该有具体值，通过类名加以访问，所有实例共用一份

class Emp{

ename:string = null; //实例属性

static companyName:string = '达内'; //静态属性

}

Emp.companyName = '达内科技';

console.log(Emp.companyName);

var e2 = new Emp();

e2.ename = 'Tom';

console.log(e2.ename);

|  |
| --- |
| Vue.js中的组件间通信：  (1)父子间通信  1)Props Down, Events Up  Parent: {  template: '...<child :child-name="myName"/>...'  data(){return { myName: '苍茫大地' }}  }  Child:{  props: ['childName']  }  ----------------------------------------------------------------  Parent: {  template: '...<child @childevent="doEvent"/>...'  methods: { doEvent(data){ data就是子组件传递的数据 } }  }  Child:{  data(){return { userInput: '' }}  ...  this.$emit('childevent', this.userInput);  }  2)$refs 和 $parent  (2)兄弟间通信  1)bus机制 |

2.补充：组件间通信

提示：Angular中兄弟组件间通信一般使用Service对象。父子间通信采用“**Props Down，Events Up**”机制。

进一步细节参见：https://angular.cn/guide/component-interaction

**父组件数据传递给子组件：**

@Component({ template: '<child [childName]="myName">..' })

class Parent{

myName = '苍茫大地';

}

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

@Component({ template: '<p>{{childName}}</p>' })

class Child{

@Input( ) //该装饰器说明这里是一个“输入型属性”

childName = null;

}

练习：创建组件 ng g component MySpace，包含数据：myName="苍茫大地"

练习：创建组件 ng g component MyPhoto，包含数据：childName，用于父组件MySpace，并将父组件的myName传递给子组件

**子组件数据传递给父组件：**

@Component({

template: '<child (onChangeMyName)="doChange(**$event**)">..' })

class Parent{ doChange(data){ data就是子组件传递的数据 } }

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

@Component({ })

class Child{ userInput = '';

@Output( ) //输出型属性

onChangeMyName = new EventEmitter( ); //事件发射器

...

this.onChangeMyName.emit( this.userInput );//发射事件

}

练习：创建组件 ng g component MyInput，包含数据：userInput；以及自定义事件发射器onChangeMyName。当用户点击“修改我的签名”时发射事件把用户输入的内容传递给父组件

3.补充：自定义模块

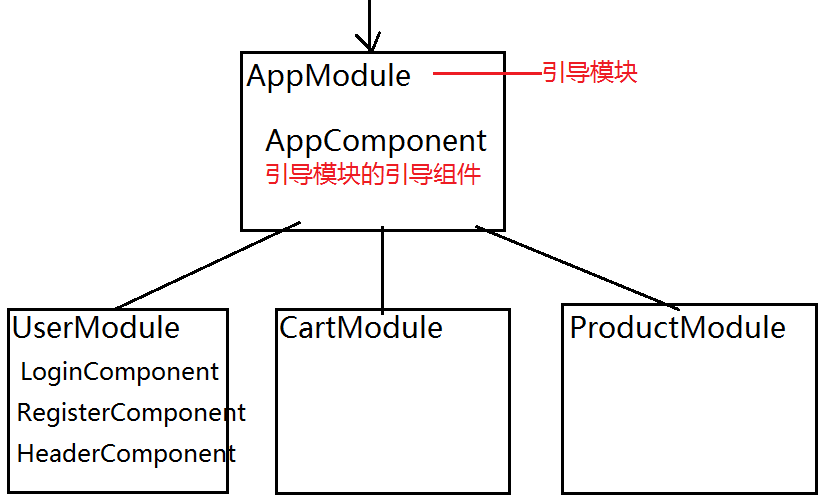
Angular核心概念1：

NgModule：模块，与ES/Node中的模块不同，不是一个文件，用于封装组件、指令、管道、服务等对象。NG中模块的概念与项目中的“功能模块”一一对应，例如学子商城：

ProductModule——商品模块：首页展示、商品列表、商品详情

UserModule——用户模块：注册、登录、收藏夹、维护注册信息、密码

CartModule——购物车模块：添加、修改、下单



每个项目中都至少有一个引导启动模块：AppModule（app.module.ts），用户还可以自定义其它模块。

如何自定义模块：

@NgModule({

declarations: [ ], 声明内部组件、指令、管道

providers: [ ], 声明内部服务对象

imports: [ ], 声明当前模块使用到了哪些其它模块

exports: [ ], 声明当前模块内部的哪些对象导出给外界

bootstrap: [ ] 声明当前模块内部的引导组件(只有引导模块有)—只有根模块的引导组件可以用于index.html

})

export class UserModule{ }

创建自定义模块： ng g module **user**

在自定义模块中创建组件： ng g component **user/**login

提示：自定义模块中创建的组件如果希望被其它模块使用，必须做到：

(1)在当前模块中声明 declarations: [ LoginComponent ]

(2)在当前模块中导出 exports: [ LoginComponent ]

(3)在其它模块中导入当前模块 imports: [ UserModule ]

午间练习：创建自定义模块UserModule，其中声明内部模块HeaderComponent，以及可供其它模块使用的组件：LoginComponent和RegisterComponent

|  |
| --- |
| 面试题：  function f1(){  var result = f2( );  console.log(result); //undefined  }  function f2(){  setTimeout(function(){  return 999;  }, 0) 指定的时间间隔到后，把回调函数放到下一次事件循环队列开头——JS解释器执行完当前队列中所有的任务后再去执行该任务  console.log('f2执行完成');  } |
| 结论：若f1调用f2；但f2的返回值是在异步方法中返回的，f1无法获取到该返回值。若就希望在f1中输出f2的异步返回值，解决方案有三种：  **(1)f1提供回调函数给f2 —— 会产生“回调地狱”**  function f1(){  f2( function(data){ console.log(data)} );  }  function f2( fn ){  setTimeout(function(){  fn( 999 );  }, 0)  }  **(2)Promise**  function f1(){  var p = f2();  p.then( fn ).catch( fn );  }  function f2( ){  return new Promise( f1, f2 ){  setTimeout(function(){  if()f1(999);  else f2(888);  }, 0)  }  }  **(3)Observable —— Angular解决返回异步数据的方案**  function f1(){  var obs = f2();  obs.subscribe((data)=>{ //订阅“可被关注的对象”  console.log(data);  })  }  function f2( ){  return new Observable( f1 ){ //返回一个“可被关注/订阅”对象  setTimeout(function(){  f1(999)  }, 0)  }  } |
| Vue.js中没有异步访问的问题：  var ProductList = {  data(){return { list: [ ] }},  mounted(){  axios.get(url).then((res)=>{  this.list = res.data; //数据访问和数据绑定都放在组件中  })  }  } |

4.Angular中实现异步访问

作用：用于异步请求服务器端数据

提示：根据Angular的设计原则，数据访问应该放在Service中，数据绑定呈现应该放在Component中 —— Angular采用的Observable解决方案。

|  |
| --- |
| H5标准没有Observable技术，它是由第三方工具Rx.js提供的。  https://cn.rx.js.org/ |

Angular中实现异步XHR请求的方法：

(1)在主模块中引入HttpClientModule

@NgModule({

imports: [ HttpClientModule ]

})

(2)创建数据库访问Service对象

class ProductService{

constructor( private http : HttpClient ){ }

getProductList(){

var observable = this.http.get( 'http://....' )

return observable;

}

}

(3)创建需要数据绑定的Component对象

class ProductListComponent{

list = null;

constructor(private s: ProductService){ }

onOnInit(){

this.s.getProductList().subscribe((data)=>{}, (err)=>{})

}

}

|  |
| --- |
| Vue.js使用VueRouter实现单页应用的步骤：  (1)视图声明路由插槽：<router-view></router-view>  (2)配置路由词典  var routes = [  {path: '/login', component: LoginComponent}  {path: '/\*', component: PageNotFoundComponent}  ]  (3)创建路由器对象  var router = new VueRouter({ routes })  (4)在根组件中使用路由器对象  new Vue({  router  }) |
| 访问测试： http://127.0.0.1:8080/index.html#/login |
| Vue.js中的路由跳转：  (1)HTML中： <router-link to="/login">跳转</router-link>  (2)JS中： this.$router.push('/login')/back()/forward().... |

5.Angular中SPA和Router

使用Angular官方提供的RouterModule(@angular/router)可以实现Single Page Application(单页应用)。

手册：https://angular.cn/guide/router

使用步骤：

(1)在根组件中声明路由插槽/出口

<router-outlet></router-outlet>

(2)在根模块中配置路由词典

var **routes** = [

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

{path: '\*\*', component: PageNotFoundComponent }

]

(3)在根模块中使用路由词典

imports: [

RouterModule.forRoot(**routes**)

]

(4)客户端访问测试

http://127.0.0.1:4200/index

注意：

1)Angular中的路由地址不能以 / 开头

2)匹配任意路由地址用 \*\*，而不是 /\*

3)客户端访问时不能在URL中出现index.html，也无需书写 #/xxx —— 最新版本NG利用H5 history.pushState工具实现了此特性

4)可以在一个路由组件中使用嵌套路由

5)在不同的路由视图间跳转有两种方式：

方式1：在HTML中

<a routerLink="/login">跳转</a>

方式2：在JS中

this.router.navigateByUrl('/login');

this.location.back()/forward()

6)路由地址中可以携带路由参数(Route Parameter)

1)声明路由地址时需要声明路由参数

{path: 'pdetail/:lid', component: ....}

2)路由跳转时需要提供路由参数值

http://127.0.0.1:4200/pdetail/17

3)组件中通过“当前激活路由”对象读取路由参数

this.activatedRoute.params.subscribe((data)=>{

data就是路由参数数据：{ lid: "17" }

})

练习：在index跳转到plist，再跳转到pdetail；从index跳转到ucenter，在ucenter的myinfo和securitymanagement间跳转

6.Angular阶段项目——“学子商城”前台重构

重构目标：“用数据绑定代替DOM操作”+“组件式开发”+ “单页应用”。

实现步骤：

(1)创建Angular脚手架项目(Node+Webpack+Angular+TS编译器)

ng new xz\_v2

(2)创建必需路由组件，配置路由词典，使得路由访问生效

(3)详细完成每个路由组件

首页组件：ng g component index

商品列表组件：ng g component product-list

商品详情组件：ng g component product-detail

没有找到组件：ng g component not-found

用户中心组件：ng g component user-center

我的信息组件：ng g component my-info

安全管理组件：ng g component security-management

课后任务：

(1)高阶面试题：Promise和Observable的区别有哪些？

(2)继续完善“xz\_v2”，试着创建ProductService，包含三个方法：getIndexData()、getProductListData()、getProductDetailData()，分别为IndexComponent、ProductListComponent、ProductDetailComponent三个组件提供异步的数据访问支持。