复习：

见思维导图

练习：创建员工列表数组： empList = [

{eid:101, ename:'亮亮', salary:5000, birthday:0, sex:1, zzmm:10},

{eid:102, ename:'然然', salary:6000, birthday:0, sex:0, zzmm:20},

{eid:103, ename:'东东', salary:7000, birthday:0, sex:1, zzmm:30},

{eid:104, ename:'涛涛', salary:8000, birthday:0, sex:0, zzmm:20},

]，把上述数据绑定在一个TABLE中，最后添加一列：<button>删除</button>，点击后可以删除该员工

1.Angular核心概念之五 —— 过滤器

Filter：过滤器，用于在View中呈现数据时显示为另一种格式；过滤器的本质是一个函数，接收原始数据转换为新的格式进行输出： function(oldVal){ 处理 return newVal }

使用过滤器： **{{ e.salary | 过滤器名 }}**

Angular2.x中，过滤器更名为“管道(Pipe)”

自定义管道的步骤：

**①创建管道class，实现转换功能**

@Pipe({ name: 'sex' })

export class SexPipe{

transform(val){ return ...}

}

②在模块中注册管道

//app.module.ts

declarations: [ SexPipe ]

③在模板视图中使用管道

{{e.empSex | sex}}

<p [title]="empSex | sex">

说明：管道的transform方法除了val还可以接收其它参数，调用管道时用冒号为这些参数赋值，例如：{{ e.empSex | sex:'en' }}

**创建管道对象的简便工具： ng g pipe 管道名**

练习：创建一个政治面貌转换过滤器/管道ZzmmPipe，可以接收一个可选参数：format，转换规则如下：

如果format值为'short'，则转换规则为：

10-党员 20-团员 30-群众

如果format值为'long'，则转换规则为：

10-中国共产党党员 20-中国共青团团员 30-中国人民群众

Vue.js中没有预定义管道；Angular提供了几个预定义管道：

①lowercase：转换为小写

{{ename | lowercase}}

②uppercase：转换为大写

{{ename | uppercase}}

③titlecase：转换为首字母大写

{{ename | titlecase}}

④slice：只显示字符串的一部分

{{ename | slice : 0 : 3 }}

⑤json：把JS对象序列化为JSON字符串

{{obj | json}}

⑥number：把数字转换为具有指定整数位和小数位的字符串

{{num | number }}

{{num | number : '4.1-4' }}

⑦currency：把数字转换为货币格式字符串：货币符号+三位一逗号+二位小数位

{{num | currency }}

{{num | currency:'￥' }}

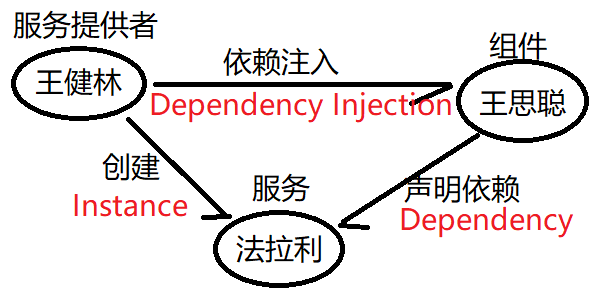
⑧date：把数字转换为日期字符串

{{num | date:'yyyy-MM-dd HH:mm:ss'}}

2.创建对象的两种方式

**方式1：手工创建式**  —— 自己创建：let c2 = new Car( )

**方式2：依赖注入式** —— 无需自己new，只需要声明依赖；服务提供者就会创建被依赖的对象，注入给服务需要者



3.Angular核心概念之六 —— 服务和依赖注入 —— 抽象&重点！

**Service：**服务，Angular认为：组件是与用户交互的一种对象，其中的内容都应该与用户操作有关系的；而与用户操作无关的内容都应该剥离出去，放在“服务对象”中，为组件服务；例如：日志记录、计时统计、数据服务器的访问.....

创建服务对象的步骤：

**①创建服务对象并指定服务提供者**

@Injectable({ providedIn: 'root' })

export class LogService{ }

**②在组件中声明依赖，服务提供者就会自动注入进来，组件直接使用服务对象即可**

export class XxComponent{

constructor(log: LogService){

//此处的log变量就会被注入为LogService实例

}

}

|  |
| --- |
| 使用服务对象时(依赖注入)常见错误：  ERROR NullInjectorError:  StaticInjectorError(AppModule)[Myc03Component -> LogService]:  StaticInjectorError(Platform: core)[Myc03Component -> LogService]:  NullInjectorError: No provider for LogService!  指定的服务对象没有服务提供者；  解决方案：  ①保证创建服务对象时指定providedIn:'root'  ②组件声明依赖时服务对象不能写错 |

|  |
| --- |
| 面试题：前端有哪些异步请求工具？各自的利弊？  ①原生XHR  ②jQuery.ajax()  ③Axios  ④Angular HttpClient  ⑤Fetch |

4.使用Angular官方提供的服务对象——HttpClient Service

HttpClient服务对象用于向指定的URL发起异步请求，使用步骤：

①在主模块中导入HttpClient服务所在的模块

//app.module.ts

imports: [ BrowserModule, FormsModule, HttpClientModule ]

②在需要使用异步请求的组件中声明依赖于HttpClient服务对象，就可以使用该对象发起异步请求了

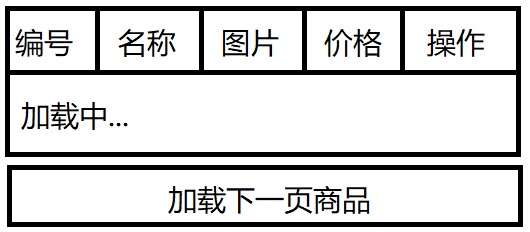
http = null

constructor( http: HttpClient ){

this.http = http

}

课后练习：使用Angular提供的异步请求服务HttpClient，完成“商品列表”功能



(1)API：http://www.codeboy.com/data/product/list.php?pno=2&kw=

(2)Angular提供了一个网络访问**服务**： HttpClient

1)在主模块中引入HTTPClientModule模块

import { HttpClientModule } from '@angular/common/http'

....

imports: [ HttpClientModule, ],

2)在组件中注入依赖的HTTPClient服务

import { HttpClient } from '@angular/common/http';

class XxxComponent{

http = null;

constructor( http: HttpClient ){ //依赖声明

this.http = http;

}

//加载商品

loadProduct(){ //3)在组件方法中使用被注入的httpClient服务

~~this.http.get(url).then( (res)=>{ } )~~

this.http.get(url).subscribe( (res)=>{ } )

}

}