Angular

主讲：陈华旺

十八、DI

- Dependency Injection 依赖注入
- 在一个类被创建为对象时，不关心 构造中传入的参数到底如何被创建对象，只关心当前类创建时该对象的传入
- DI 的使用
    - 1、定义一个普通的类
    - 2、在 `app.model.ts` 文件中，将该类定义为注入提供者
    - 3、在需要组件的构造函数中，定义该类的对象名称，完成依赖注入

app.model.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { FormsModule, ReactiveFormsModule } from '@angular/forms'

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';

import { DiComponentComponent } from './componets/dayFive/DI-components/di-component/di-component.component'
import { DataService } from './componets/dayFive/DI-components/dataService';
import { DiComponentChildComponent } from './componets/dayFive/DI-components/di-component-child/di-component-child.component';


@NgModule({
  declarations: [
    AppComponent,
    DiComponentComponent,
    DiComponentChildComponent

  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    // 模板表单
    FormsModule,
    ReactiveFormsModule
  ],
  // 提供者 用于为当前项目中所有需要注入参数的 组件 提供 对象的 指定
  providers: [
    DataService
  ],
  // bootstrap: [AppComponent,NewoneComponent]
  bootstrap: [AppComponent]
})
export class AppModule { }
```

其它组件

```
import { Component, OnInit } from '@angular/core';
import { DataService } from '../dataService';

@Component({
  selector: 'app-di-component',
  templateUrl: './di-component.component.html',
  styleUrls: ['./di-component.component.css']
})
export class DiComponentComponent implements OnInit {

  // private ds = new DataService();

  //被注入的组件会作为，该组件的属性存在
  constructor(private ds:DataService) { }

  ngOnInit() {
  }
}
```

- angular 中注入的特性
  - 1、被作为注入类的类（提供者），只会被创建一次
  - 2、被创建的注入类可以为多个组件提供服务
  - 3、因为对象只有一个，可以使用依赖注入完成 组件间 数据传递（中央数据总线）
  - 4、注入的 搜索方式
- 主要目的，做解耦合，和MVC 的 Service 分离

十九、HTTP服务

- 远端数据获取，angular 项目 只能是用 异步请求（AJAX）方式获取远端数据

19.1、导入 HTTP 模块

- 在app.module.ts文件中加载模块

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { FormsModule, ReactiveFormsModule } from '@angular/forms'
import { HttpModule,JsonpModule } from '@angular/http';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';

import { HttpAjaxComponent } from './componets/dayFive/http-
components/http-ajax/http-ajax.component';

@NgModule({
  declarations: [
    AppComponent,
    HttpAjaxComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    // 模板表单
    FormsModule,
    ReactiveFormsModule,
    // 同服务器的正常请求
    HttpModule,
    // 跨服务的跨域请求
    JsonpModule
  ],
  // 提供者 用于为当前项目中所有需要注入参数的 组件 提供 对象的 指定
  // 自定义的 对象 提供
  providers: [
    DataService
  ],
  // bootstrap: [AppComponent,NewoneComponent]
  bootstrap: [AppComponent]
})
export class AppModule { }
```

19.2、配置代理服务

- 在项目跟目录下创建一个 `proxy.cfg.json` 文件

```
{
    "/api":{
        "target":"http://127.0.0.1:80"
    }
}
```

- 启动服务时 以 代理模式启动

```
ng server --proxy-config proxy.cfg.json
```

- 或者在package.json 文件中 进行 定义启动项

```
"scripts": {
    "ng": "ng",
    "start": "ng server --proxy-config proxy.cfg.json",
    "build": "ng build",
    "test": "ng test",
    "lint": "ng lint",
    "e2e": "ng e2e"
  }
```

```
npm start
```

19.3、Http.get() 方法使用

数据获取

```
import { Component, OnInit } from '@angular/core';
import { Http ,URLSearchParams ,Jsonp } from '@angular/http';
import { FormGroup, FormControl } from '@angular/forms';

@Component({
  selector: 'app-http-ajax',
  templateUrl: './http-ajax.component.html',
  styleUrls: ['./http-ajax.component.css']
})
export class HttpAjaxComponent implements OnInit {

  private user = {};

  constructor(
    private http:Http,
    private jsonp:Jsonp
  ) { }

  ngOnInit() {
  }

  // 单纯数据获取
  private getData(){
    // let response = this.http.get("http://127.0.0.1:80/angularData.php");
    let response = this.http.get("/api/angularData.php");
    //  0 1 2 3 4

    // 订阅
    //    当前请求 完全结束 并且获取数据后 进行数据处理
    response.subscribe((data)=>{
      // 对数据进行解析
      //  data ==> Response
      //      存在一个方法  json() 获取AJAX请求的返回数据
      // console.log(data.json());
      this.user = data.json()
    });
  }
}
```

传递数据

```
import { Component, OnInit } from '@angular/core';
import { Http ,URLSearchParams ,Jsonp } from '@angular/http';
import { FormGroup, FormControl } from '@angular/forms';

@Component({
  selector: 'app-http-ajax',
```

```
  templateUrl: './http-ajax.component.html',
  styleUrls: ['./http-ajax.component.css']
})
export class HttpAjaxComponent implements OnInit {

  private userForm = new FormGroup({
    name:new FormControl("",[]),
    age:new FormControl("",[])
  })

  constructor(
    private http:Http,
    private jsonp:Jsonp
  ) { }

  ngOnInit() {
  }

  private sendData(){
    let url = "/api/angularSendData.php";
    /*
      {
        url?: string|null
        method?: string|RequestMethod|null
        search?: string|URLSearchParams|{[key: string]: any | any[]}|null
        params?: string|URLSearchParams|{[key: string]: any | any[]}|null
        headers?: Headers|null
        body?: any
        withCredentials?: boolean|null
        responseType?: ResponseContentType|null
      }

      {
        key:value,
        key:value
      }

     */
    // URLSearchParams  需要导包，但是 VSCode 不提示导包
    // let prams = new URLSearchParams();
    // prams.append("name",this.name);
    // prams.append("age",this.age);

    // this.http.get(url,{
    //   // params:prams
    //   search:prams
    // }).subscribe((data)=>{
    //   console.log(data);
    // })
```

```
    console.log(this.userForm.value);

    this.http.get(url,{
      // params:prams
      // search:{
      //   name:this.name,
      //   age:this.age
      // }
      search:this.userForm.value
    }).subscribe((data)=>{
      console.log(data);
    })
  }
}
```

19.4、Http.post() 方法使用

```
import { Component, OnInit } from '@angular/core';
import { Http ,URLSearchParams ,Jsonp } from '@angular/http';
import { FormGroup, FormControl } from '@angular/forms';

@Component({
  selector: 'app-http-ajax',
  templateUrl: './http-ajax.component.html',
  styleUrls: ['./http-ajax.component.css']
})
export class HttpAjaxComponent implements OnInit {

  private userForm = new FormGroup({
    name:new FormControl("",[]),
    age:new FormControl("",[])
  })

  constructor(
    private http:Http,
    private jsonp:Jsonp
  ) { }

  ngOnInit() {
  }

  private sendDataPost(){
    let url = "/api/angularSendData.php";
    // this.http.post(url,body,optionArgs)
    // body 是请求需要传递的数据
    // URLSearchParams  用在POST 请求中  FromsData 类型的数据传递
    let params = new URLSearchParams();
    params.append("name",this.userForm.value.name);
    params.append("age",this.userForm.value.age);
    this.http.post(url,params)
        .subscribe((data)=>{
          console.log(data.json());
        });
  }
}
```

19.5、Jsonp.jsonp() 方法使用

```
import { Component, OnInit } from '@angular/core';
import { Http ,URLSearchParams ,Jsonp } from '@angular/http';
import { FormGroup, FormControl } from '@angular/forms';

@Component({
  selector: 'app-http-ajax',
  templateUrl: './http-ajax.component.html',
  styleUrls: ['./http-ajax.component.css']
})
export class HttpAjaxComponent implements OnInit {

  private wk:string="";

  constructor(
    private http:Http,
    private jsonp:Jsonp
  ) { }

  ngOnInit() {
  }

  private seachData(){
    // 跨域请求
    let url = "https://sp0.baidu.com/5a1Fazu8AA54nxGko9WTAnF6hhy/su";
    // let params = new URLSearchParams();
    // params.append("wd",this.wk);
    // params.append("age",this.userForm.value.age);
    // JSONP 方法 只能使用 get请求
    // this.jsonp.post(url,params)
    //     .subscribe((data)=>{
    //       console.log(data.json);
    //     });

    this.jsonp.get(url,{
      params:{
        "wd":this.wk,
        // JSONP_CALLBACK 关键占位符，以字符的方式定位在请求中
        //   格式 大小写 都不能变
        "cb":"JSONP_CALLBACK"
        // key 是服务器所需要的回调名
        // value "JSONP_CALLBACK"
      }
    }).subscribe((data)=>{
      console.log(data.json());
    });
  }
}
```

二十、路由

二十一、项目打包和发布