# Angular 服务与Rxjs

# 服务简介

## 服务的概念

服务是指为他人做事，并使他人从中受益的一种有偿或无偿的活动。不以实物形式而以提供劳动的形式满足他人某种特殊需要。在Angular中，服务用于帮助开发者书写可重用的公共功能（如日志处理、权限管理等）和复杂业务逻辑，对于应用程序的模块化有着很重要的意义。

## Angular服务

Angular服务一般是封装了某种特定功能的独立模块，它可以通过依赖注入的方式供外部调用。服务在Angular中使用十分广泛，例如:

1. 多个组件中出现重复代码时，把重复代码提取到服务中实现代码服用。
2. 当组件中掺杂了大量的业务代码和数据处理逻辑时，把这些逻辑封装成服务供组件使用，组件只负责UI相关的逻辑，有利于后续的更新和维护。
3. 把需要共享的数据存储在服务中，通过在多个组件中注入同一服务实例实现数据共享。

# 共享服务

在前面的课程中我们已经介绍了组件间通讯的几种方法，路由（非常简便）、@Input（逻辑比较严谨）、共享服务（场景广泛，局限性低）。但在实际开发中，通常需要在多个组件之间进行通信，这种情况下可以通过组件间共享同一服务实例（通过依赖注入同一层级的服务）来实现通信

# HTTP服务

HTTP服务是Angular中使用HTTP协议与远程服务器进行通讯的一个独立模块HttpModule。在Angular应用中使用HTTP服务只需要三个简单步骤：

1. 在根模块模块装饰器@NgModule中导入HttpModule。
2. 在组件模块中导入HTTP服务。
3. 在组件的构造函数中声明引入。（依赖注入）

引入HTTP服务后，组件就可以用AJAX和JSONP两种方式发送数据请求。

## AJAX介绍

AJAX（Asynchronous JavaScript and XML）是使用XMLHttpRequest（XHR）对象向服务器发送请求处理响应的通讯技术。XMLHttpRequest默认采取异步的方式发送请求。

为什么默认采取异步：我们不能阻塞发送请求后服务器处理时，UI的响应。

一般来说有以下三种方式处理异步操作。

1. 使用回调函数。

$.ajax({ success: function( ….. )}) 回调地狱。

1. 使用Promise。

Es6特性， then() yield… 来解决回调地狱的问题。

1. 使用Observable。

Angular中默认的处理异步的方式。

## HTTP GET请求

什么情况叫跨域。URI的协议（Scheme）/主机名（Host）/端口（port）这三个部分完全相同才算是同源。

解决跨域的思路：

1. 服务器端解决。 Java Nginx，nodejs node-http-proxy
2. 客户端解决（浏览器端）在Angular里我们用代理的方式。

步骤：

1. 在项目根路径下创建proxy.conf.json文件。
2. 在package.json中配置"start": "ng serve --proxy-config proxy.conf.json"。

## HTTP POST请求

## HTTP JSONP请求

实际上是利用<script>标签跨域特性。

注意：要在根模块里引入JsonpModule 模块

## HttpInterceptor示例（服务拦截）

主要就是教大家怎么样着手去修改源码。

具体步骤看视频。

# WS服务 （课次下）

WebSocket协议是基于TCP的一种新的网络协议。它实现了浏览器与服务器全双工(full-duplex)通信——允许服务器主动发送信息给客户端。

# Rxjs简介（课次下）

在Angular中，可以使用可观察对象作为数据架构的骨架来构建应用，使用可观察对象构造数据被称为响应式编程（Reactive Programming）。

数据架构要考虑的问题：

1. 如何将所有不同的数据源聚合成一个完整的体系。
2. 如何防止意想不到的副作用导致的BUG。Try catch
3. 如何更好地构建代码以使其更容易维护并且让新来的团队成员更容易上手。
4. 当数据发生变化时，能够让应用尽快响应。 MVW（双向数据绑定）

在维基百科中的解释是：一种面向数据流（data Flows）和变化传播（the Propagation of Change）的编程范式。这意味着可以在编程语言中很方便地表达静态或动态的数据流，而相关的计算模型会自动将变化的值通过数据流进行传播。

ReactiveX（Reactive Extensions），一般简写为Rx，它是微软开发并维护的基于Reactive Programming范式实现的一套工具库集合，于2012年11月开源，用于提供一系列接口规范来帮助开发者更方便地处理异步数据流。Rx系列结合了观察者模式、迭代器模式和函数式编程。

官网：<http://reactivex.io/rxjs/>

响应式编程跟传统的命令式编程相比较：

A =1 ; b= 2;

A + B = C;

C ? => 3

A = 2;

C ? => 在命令式编程里C还是等于3，而在响应式的数据流中C =4 。

我在 mvvm 里当model变化的时候View也会改变，体现出了面向变化传播的思想。

## Observable介绍

在Rx中，最核心的概念就是Observable。操作中产生的异步数据都需要先包装成Observable对象，Observable对象的作用是把这些异步的数据变换为数据流形式。所以生成的这些Observable对象相当于数据流的源头，后续的操作都是围绕着这些被转换的流动数据展开。Observable作为观察者模式中的被观察者角色。

定义一个 Observable

1、什么时候发射一个元素。Next

2、什么时候抛出异常。Error

3、什么时候发出结束流信号 Complete

在Rx的理念中，Observable有冷、热模式。默认是冷模式，只有当subscribe的时候才开始执行。

Observable跟Promise的区别：

Promise

1、返回单个值

2、不可取消的

Observable

1、随着时间的推移发出多个值

2、可以取消的

3、支持 map、filter、reduce 等操作符

4、延迟执行，当订阅的时候才会开始执行

## Operator介绍（最关键的）

Rx在结合了观察者模式的同时，还结合了函数式编程和迭代器的思想。其中，Rx的Operator便是对这两种编程模式的具体体现。

Operator，顾名思义，是Rx中Observable的操作符。在Rx中，每一个Observable对象，或者说数据流，都可以通过某个operator对该Observable对象进行变换、过滤、合并和监听等操作。同时，大多数的operator在对Observable对象处理后会返回一个新的Observable对象供下一个operator进行处理。这样方便在各个operator之间通过链式调用的方式编写代码。操作符分为以下几种：

1. 创建操作符：例如Observable.fromEvent() 和 new Observable()、Observable.of()等。
2. 变换操作符：最常见的 Observable.map()等
3. 过滤操作符：Observable.filter()等
4. 组合操作符：Observable.forkJoin()等
5. 工具操作符：Observable.delay()、Observable.timeout()等

## Subscrible简介

在Rx中subscribe()方法会用来订阅Observable对象发出的所有事件，其接收一个observer作为参数，来对observable发出的事件进行订阅。每当observable完成并发送（Emit）一个事件时，该事件就会被observer所捕获，进入到observer对应的回调函数中。

## Observer简介

Observer对Obervable对象发出的每个事件进行响应。该对象中包含 next、error 和 complete 方法。

## Subscription 简介

Observable对象被订阅后返回的实例。

## Subject 简介

EventEmitter的等价数据结构，Subject 既是 Observable 对象，又是 Observer 对象

“Subject看成是一个Observable的集合”当有新消息时，Subject 会对内部的 observers 列表进行组播 (multicast)

有什么好处：

场景：当用户在一个文本输入框进行输入的时候，需要根据用户的实时输入进行监听，与服务器进行交互。（类似于淘宝搜索）。

限制：

1. 不必要用户每次输入的时候都发送请求，间隔500
2. 在向服务器发送请求前检查输入内容，如果内容相同那么就不再发送了。
3. 要保证请求返回的顺序。

以上内容用原生的javascript去实现就会比较复杂了。

而用observable来做so easy. 如下：

let inputSelector = document.querySelector('input');

Rx.Observable.fromEvent(inputSelector,'keyup').debounceTime(500).//间隔500

distinctUntilChanged()//值改变才往下走.switchMap(....)//拦截阻断当前的事件.subscribe(callback);

# 最新的HTTPClient

在Angular 4.3.0-rc.0 版本之后出现了HttpClientModule模块，这个模块是HttpModule功能的升级。详情请见官网：<https://www.angular.cn/guide/http>

其主要特点如下：

## 默认 JSON 解析

现在 JSON 是默认的数据格式，我们不需要再进行显式的解析。即我们不需要再使用以下代码：

http.get(url).map(res => res.json()).subscribe(...)

现在我们可以这样写：

http.get(url).subscribe(...)

## 支持拦截器 (Interceptors)

拦截器允许我们将中间件逻辑插入管线中。

拦截有什么用：

1. 需要给全局的HTTP请求定义一个公共的请求头。
2. 可以对某个特定的错误进行统一处理（401错误，没权限）。
3. 可以在请求发出前来一个 loading.. 在返回结果后关闭这个loading…

………

## 进度事件 (Progress Events)

进度事件可以用于跟踪文件上传和下载。