# 框架

面向对象：

面向对象是一种程序开发的方法，它将对象作为程序的基本单元，将程序和数据封装其中，以提高软件的灵活性、重用性和扩展性。对象是把数据及对数据的操作方法放在一起，作为一个相互依存的整体。再说一下类与对象，类描述了一组有相同特性和相同行为的对象，具有相同属性和相同方法的对象的抽象就是类。即，对象的抽象是类，类的实例是对象。在面向对象的编程中，把用类创建对象的过程称为实例化。

面向过程与面向对象的区别

面向过程是一种直接的编程方法，它是按照编程语言的思路考虑问题。通过顺序执行一组语句来实现一个功能，这些语句的执行过程就是整个程序。

面向对象是一种抽象度更高的编程方法。它的目标是使模块的抽象度更高，目的是可复用。面向对象三大特性：封装，继承，多态。

面向过程可以说是从细节方面思考问题

面向对象可以说是从宏观方面思考问题

对象的创建方式有对象字面量，工厂方式，原型方式，构造函数，混合模式。

原型，原型链：

我们创建的每个函数都有一个prototype属性，这个属性是一个指针，指向一个对象，这个对象的用途是包含可以由特定类型的所有实例共享的属性和方法。那么，prototype就是通过调用构造函数而创建的那个对象实例的原型对象。

使用原型的好处是可以让对象实例共享它所包含的属性和方法。也就是说，不必在构造函数中添加定义对象信息，而是可以直接将这些信息添加到原型中。使用构造函数的主要问题就是每个方法都要在每个实例中创建一遍。

在JavaScript中,一共有两种类型的值,原始值和对象值。每个对象都有一个内部属性 prototype ,我们通常称之为原型。原型的值可以是一个对象,也可以是null。如果它的值是一个对象，则这个对象也一定有自己的原型。这样就形成了一条线性的链，我们称之为原型链。

简单的说就是在JavaScript 中，每当定义一个对象（函数）时候，对象中都会包含一些预定义的属性。其中函数对象的一个属性就是原型对象 prototype。JS在创建对象（不论是普通对象还是函数对象）的时候，都有一个叫做\_\_proto\_\_的内置属性，用于指向创建它的函数对象的原型对象prototype，我们把这个有\_\_proto\_\_串起来的直到Object.prototype.\_\_proto\_\_为null的链叫做原型链

angularjs：

angular是谷歌开发的一款js框架，最核心的机制之一是数据双向绑定，利用脏检查机制，原理是Angular 在 scope 模型上设置了一个 监听队列，用来监听数据变化并更新 view 。每次绑定一个东西到 view 上时 AngularJS 就会往 $watch 队列里插入一条 $watch ，用来检测它监视的 model 里是否有变化的东西。当浏览器接收到可以被 angular context 处理的事件时，$digest 循环就会触发，遍历所有的 $watch ，最后更新 dom。Angular还提供了很多服务供我们使用（可简单回答下题服务），另外利用依赖注入在控制器中使用，说到依赖注入它是一种软件设计模式，目的是处理代码之间的依赖关系，减少组件间的耦合。Angular里面还有过滤器，例如orderby，filter等帮助我们变换输出形式，它还提供了自定义过滤器，根据过滤器参数的不同我们可以根据实际的需求做不同的返回处理，angular里面还有自定义指令帮助我们在构建动态web应用时弥补静态文本的不足，还提供了很多插件，例如动画插件，路由插件，等，路由可以让我们开发spa应用（可回答路由）

angular服务：

angular里面内置了三十多种服务，最常用的就是$http用来前后数据交互的，其次就是一些location，$q等服务，除了内置服务，angular还提供了五种自定义服务供我们使用，有constant

，value，constant和value主要就是用于存放一些数据或方法以供使用,区别是constant一般是存放固定内容,value存放可能会被修改的内容，然后还有factory，它和constant,value最大的区别是,factory服务是有一个处理过程,经过这个过程,才返回结果的. 还有service方法，ervice和factory的区别在于,它第二个参数传入的是一个构造函数,最后被注入的服务是这个构造函数实例化以后的结果.所以基本上使用service创建的服务的,也都可以使用factory来创建，还有一种就是provider，它属于供应商服务，设置里面必须先返回一个$get的构造函数，然后再返回值，这种写法就是可以在config中对设置的数据进行二次修改。后三种是我们比较常用的服务，他们三者之间的关系大致是这样的factory 把 service 的方法和数据放在一个对象里，并返回这个对象；service 通过构造函数方式创建 service，返回一个实例化对象；provider 创建一个可通过 config 配置的 service。

angular路由：

Angular1.x 中常用 ngRoute 和 ui.router，还有一种为 Angular2 设计的 [new router](https://angular.github.io/router/) （面向组件），ngRoute 模块是 Angular 自带的路由模块，而 ui.router 模块是基于 ngRoute模块开发的第三方模块。ui.router 是基于 state （状态）的， ngRoute 是基于 url 的，ui.router模块具有更强大的功能，主要体现在视图的嵌套方面。使用 ui.router 能够定义有明确父子关系的路由，并通过 ui-view 指令将子路由模版插入到父路由模板的 ui-view 中去，从而实现视图嵌套。而在 ngRoute 中不能这样定义，如果同时在父子视图中 使用了ng-view 会陷入死循环。

混合开发：

混合开发就是原生＋html5构建的bybridapp，我们前端开发人员进行混合项目构建，常利用的封壳（打包app）平台有cordova，hbuilder hello5+，两个平台的区别在于cordova需要借助原生的环境，封装android类型需要有androidsdk以及java jdk环境，ios类型需要有xcorde，h5+平台的话是借助dcloud云端大包平台，可以直接封装。混合应用相对于原生应用来说有很多优点，例如一次编译多平台运行，ui表现一致，更新不需要审查等，同时也有一些缺点，在性能上不如原生，很多硬件功能没有原生完善，不过可以借助cordova。H5+提供的硬件api进行完善。还有就是前段开发人员与ios或者安卓人员进行合作我们需要开发页面，实现样式以及功能交互，然后原生人员在原生壳子中内嵌轻量级浏览器，通过webview加载我们写的页面，这就是原生根h5人员的配合开发混合项目

react：

facebook在构建instagram网站的时候遇见两个问题一个是数据绑定的时候，大量操作真实dom，性能成本太高，然后就是网站的数据流向太混乱，不好控制，于是就推出了react框架，它是一个重视view层的框架，主要特点是声明事，自动dom操作。主要利用的就是虚拟dom，主要特点是单向数据流，核心方法是组件，组件的优势在于组件将数据和逻辑封装，能够提高代码复用率，组件高内聚低耦合，很容易对单个组件进行测试。因此降低测试难度，另外就是组件使用直观的语法可以极大提高可读性，降低代码复杂度。React中提供了属性和状态进行数据的传递，还有生命周期方便对组件的操作，生命周期分为三个阶段，初始化，运行中，卸载阶段，每个阶段都提供了钩子函数供我们去管理组件的操作。利用react与webpack我们可以进行模块化开发，方便对每个组件单独管理与操作，react中还提供了路由，react－router，通过link 匹配路径，router history管理路由路径，route设置具体的路由路径，已经该路径要加载的组件，同时还提供了indexRoute方便我们设置初始加载的组件试图。React还有两个比较强大的数据流管理框架flux和redux（接下来可以回答下题）

flux，redux：

redux是一个应用数据流框架，主要是解决了组件间状态共享的问题，原理是集中式管理，主要有三个核心方法，action，store，reducer，工作流程是view调用store的dispatch接收action传入store，reducer进行state操作，view通过store提供的getState获取最新的数据，flux也是用来进行数据操作的，有四个组成部分action，dispatch，view，store，工作流程是view发出一个action，派发器接收action，让store进行数据更新，更新完成以后store发出change，view接受change更新视图。Redux和Flux很像。主要区别在于Flux有多个可以改变应用状态的store，在Flux中dispatcher被用来传递数据到注册的回调事件，但是在redux中只能定义一个可更新状态的store，redux把store和Dispatcher合并,结构更加简单清晰

新增state,对状态的管理更加明确，通过redux，流程更加规范了，减少手动编码量，提高了编码效率，同时缺点时当数据更新时有时候组件不需要，但是也要重新绘制，有些影响效率。一般情况下，我们在构建多交互，多数据流的复杂项目应用时才会使用它们

node：

node.js是一个基于Chrome v8引擎的javascript 运行环境。Node.js使用了一个事件驱动、非阻塞式 I/O的模型，使其轻量又高效。使用C＋＋语言编写，特点是快，性能高，开发效率高，应用范围广，但是也耗费内存，比较新，ied不是很完善，中间件少，node可以解析js，没有浏览器安全级别限制，同时也提供了很多系统级别的api，像文件的读取，进程的管理等，我经常使用的node里面的模块有url对地址栏进行地址的解析操作，querystring用于解析与格式化 URL 查询字符串，fs对文件进行操作，http，https模块中的request方法进行获取数据，发送数据等。。，同时node还提供了路由，可以通过加载不同url显示不同视图内容，同样可以模拟后端不同接口存储不同的数据（剩下的可以回答下面几题）

mongodb：

MongoDB是一个基于分布式文件存储的数据库。由C++语言编写。是世界上目前用的最广泛的nosql数据库，也就是非关系型数据库，他的特点是高性能、易部署、易使用，存储数据非常方便。

他有以下优点，例如面向文档储存的数据库也就是BSON格式存储，说道bson，BSON基于JSON格式，对JSON进行改造的，bson的特点就是便利更快，操作更简单，同时增加了额外的数据类型， 然后mango还具有丰富的查询指令，支持索引，具有分片系统，无模式，同时也有一些缺点占用空间大，不支持事物，目前也不维护32位系统了，一个芒果里面可以创建很多数据库，数据库事由集合组成，集合就是 MongoDB 文档组，类似于 RDBMS （关系数据库管理系统：Relational Database Management System）中的表格。那文档其实就是一个键值(key-value)对也就是BSON，常用的操作就是对集合数据的增删改查。

express：

express 是基于node平台 开发的一个极其轻量的框架 帮助你快速开发web以及移动端应用，express不对node.js本身的特性进行二次抽象 而是在基本功能上进行扩充，express完全是由路由和中间件构成的框架 从本质上来说一个express应用就是为了调用各种中间件。中间件是一个函数 --他可以访问request对象，响应对象(res) 以及处于请求--响应流程中的中间件，一般被命名为 next 的变量，中间件可以执行任何代码，可以修改请求和响应对象，可以结束请求－响应对象，也可以调用堆栈中的下一个中间件，我们把结束请求-响应流程的方法 称为是句柄函数。Express提供很多中间件，方便我们对于系统数据视图的操作与管理，与之配合的是我们常用的ejs模版引擎（答下题）

ejs：

EJS是一个简单高效的模板语言，通过数据和模板，可以生成HTML标记文本。可以说EJS是一个JavaScript库，EJS可以同时运行在客户端和服务器端，客户端安装直接引入文件即可，服务器端用npm包安装。使用ejs有很多优点。例如：可以快速编译和渲染，使用简单的模板标签，还可以自定义标记分隔符，支持文本包含，支持浏览器端和服务器端，模板静态缓存，支持express视图系统等，ejs常用流程控制标签，可以更好的解析变量，操作逻辑。

vue：

vue是一个构建数据驱动的 web 界面的渐进式框架。Vue.js 的目标是通过尽可能简单的 API 实现响应的数据绑定和组合的视图组件。Vue.js借鉴了Angular以及React的一些核心思想，综合各自的长处进行了操作以及性能等方面的优化，vue比angular简单，小，运行速快快，都有数据双向绑定，vue与react都使用 virtual DOM虚拟dom，都提供了组件化的视图组件，都有丰富的插件库，react使用jsx渲染页面，vue使用更简单的模版。Vue也提供了丰富的指令进行操作，同时也有生命周期，­­所有的生命周期钩子自动绑定 this 上下文到实例中，因此你可以访问数据，对属性和方法进行运算。Vue.js 使用了基于 HTML 的模版语法，允许开发者声明式地将 DOM 绑定至底层 Vue 实例的数据。所有 Vue.js 的模板都是合法的 HTML ，所以能被遵循规范的浏览器和 HTML 解析器解析。Vue也提供了路由操作，通过router－link设置路径，path进行路由路径配置，components进行组件模版加载。他还提供了动画插件，可以结合animate。Css进行动画设置。还有自定义指令，过滤器等。

然后就是Vuex 是一个专为 Vue.js 设计的状态管理模式，vuex解决了组件之间同一状态的共享问题。当我们的应用遇到多个组件共享状态时，会需要多个组件依赖于同一状态，这时候使用vuex就可以很好的解决。同时vue可以配合mint –ui这个ui框架快速构建移动端应用。mint –ui提供了很多漂亮的ui组件设计，方便我们快读构建开发项目。

ionic：

ionic是一个用来开发混合手机应用的，开源的，免费的代码库。可以优化html、css和js的性能，构建高效的应用程序，而且还可以用于构建Sass和AngularJS的优化。ionic是一个专注于用WEB开发技术，基于HTML5创建类似于手机平台原生应用的一个开发框架。绑定了AngularJS和Sass。这个框架的目的是从web的角度开发手机应用，基于PhoneGap的编译平台，可以实现编译成各个平台的应用程序。她有以下特点1.ionic 基于Angular语法，简单易学，是一个轻量级框架。完美的融合下一代移动框架，支持 Angularjs 的特性， MVVM ，代码易维护。ionic 提供了漂亮的设计，通过 SASS 构建应用程序，它提供了很多 UI 组件来帮助开发者开发强大的应用。ionic 专注原生，让你看不出混合应用和原生的区别，ionic 提供了强大的命令行工具。ionic 性能优越，运行速度快。

单页面应用：

单页面应用single page web application

一般我们开发单页面应用有以下几个优点：

1. 分离前后端关注点，前端负责界面显示，后端负责数据存储和计算，各司其职，不会把前后端的逻辑混杂在一起；
2. 减轻服务器压力，服务器只用出数据就可以，不用管展示逻辑和页面合成，吞吐能力会提高几倍；
3. 同一套后端程序代码，不用修改就可以用于Web界面、手机、平板等多种客户端；

除了优点同时还有一些缺点；例如SEO问题，现在可以通过Prerender等技术解决一部分；前进、后退、地址栏等，需要程序进行管理；书签，需要程序来提供支持；

* 使用angular来开发的话有以下几个特点：
* 速度：更好的用户体验，让用户在web app感受native app的速度和流畅，
* MVC：经典MVC开发模式，前后端各负其责。
* ajax：重前端，业务逻辑全部在本地操作，数据都需要通过AJAX同步、提交。
* 路由：在URL中采用#号来作为当前视图的地址,改变#号后的参数，页面并不会重载。

#### 谈谈你对webpack的看法

WebPack 是一个模块打包工具，你可以使用WebPack管理你的模块依赖，并编绎输出模块们所需的静态文件。它能够很好地管理、打包Web开发中所用到的HTML、JavaScript、CSS以及各种静态文件（图片、字体等），让开发过程更加高效。对于不同类型的资源，webpack有对应的模块加载器。webpack模块打包器会分析模块间的依赖关系，最后 生成了优化且合并后的静态资源。  
**webpack的两大特色：**

1.code splitting（可以自动完成）

2.loader 可以处理各种类型的静态文件，并且支持串联操作

webpack 是以commonJS的形式来书写脚本滴，但对 AMD/CMD 的支持也很全面，方便旧项目进行代码迁移。

webpack具有requireJs和browserify的功能，但仍有很多自己的新特性：

1. 对 CommonJS 、 AMD 、ES6的语法做了兼容

2. 对js、css、图片等资源文件都支持打包

3. 串联式模块加载器以及插件机制，让其具有更好的灵活性和扩展性，例如提供对CoffeeScript、ES6的支持

4. 有独立的配置文件webpack.config.js

5. 可以将代码切割成不同的chunk，实现按需加载，降低了初始化时间

6. 支持 SourceUrls 和 SourceMaps，易于调试

7. 具有强大的Plugin接口，大多是内部插件，使用起来比较灵活

8.webpack 使用异步 IO 并具有多级缓存。这使得 webpack 很快且在增量编译上更加快

**es5 vs es6**

es5:

### **1. strict模式**

严格模式，限制一些用法，'use strict';

### **2. Array增加方法**

增加了every、some 、forEach、filter 、indexOf、lastIndexOf、isArray、map、reduce、reduceRight方法

PS： 还有其他方法 Function.prototype.bind、String.prototype.trim、Date.now

### **3. Object方法**

Object.getPrototypeOf

Object.create

Object.getOwnPropertyNames

Object.defineProperty

Object.getOwnPropertyDescriptor

Object.defineProperties

Object.keys

Object.preventExtensions / Object.isExtensible

Object.seal / Object.isSealed

Object.freeze / Object.isFrozen

**Es6**

ECMAScript6在保证向下兼容的前提下，提供大量新特性

1.块级作用域 关键字let, 常量const

2.对象字面量的属性赋值简写（property value shorthand）

3.赋值解构

4.函数参数 - 默认值、参数打包、 数组展开（Default 、Rest 、Spread）

5.箭头函数 Arrow functions

6.字符串模板 Template strings

7. Iterators（迭代器）+ for..of

8.生成器 （Generators）

9.Class

Class，有constructor、extends、super，但本质上是语法糖（对语言的功能并没有影响，但是更方便[程序员](http://caibaojian.com/a-programmer/" \o "程序员)使用）。

10.Modules

ES6的内置模块功能借鉴了CommonJS和AMD各自的优点：

(1).具有CommonJS的精简语法、唯一导出出口(single exports)和循环依赖(cyclic dependencies)的特点。

(2).类似AMD，支持异步加载和可配置的模块加载。

11.Map + Set + WeakMap + WeakSet

四种集合类型，WeakMap、WeakSet作为属性键的对象如果没有别的变量在引用它们，则会被回收释放掉。

12.Math + Number + String + Array + Object APIs

13. Proxies

使用代理（Proxy）监听对象的操作，然后可以做一些相应事情。

14.Symbols

Symbol是一种基本类型。Symbol 通过调用symbol函数产生，它接收一个可选的名字参数，该函数返回的symbol是唯一的。

15.Promises

Promises是处理异步操作的对象，使用了 Promise 对象之后可以用一种链式调用的方式来组织代码，让代码更加直观（类似[jQuery](http://caibaojian.com/jquery/" \o "jQuery手册)的deferred 对象）。

### **一. Sass/Scss、Less是什么?**

Sass (Syntactically Awesome Stylesheets)是一种动态样式语言，Sass语法属于缩排语法，比css比多出好些功能(如变量、嵌套、运算,混入(Mixin)、继承、颜色处理，函数等)，更容易阅读。

**Sass与Scss是什么关系?**

Sass的缩排语法，对于写惯css前端的web开发者来说很不直观，也不能将css代码加入到Sass里面，因此sass语法进行了改良，Sass 3就变成了Scss(sassy css)。与原来的语法兼容，只是用{}取代了原来的缩进。

Less也是一种动态样式语言. 对CSS赋予了动态语言的特性，如变量，继承，运算， 函数.  Less 既可以在客户端上运行 (支持IE 6+, Webkit, Firefox)，也可在服务端运行 (借助 Node.js)。

### **二. Sass/Scss与Less区别**

### **1.编译环境不一样**

Sass的安装需要Ruby环境，是在服务端处理的，而Less是需要引入less.js来处理Less代码输出css到浏览器，也可以在开发环节使用Less，然后编译成css文件，直接放到项目中，也有 Less.app、SimpleLess、CodeKit.app这样的工具，也有在线编译地址。

### 2.变量符不一样，Less是@，而Scss是$，而且变量的作用域也不一样。

### 3.输出设置，Less没有输出设置，Sass提供4中输出选项：nested, compact, compressed 和 expanded。

#### 4.Sass支持条件语句，可以使用if{}else{},for{}循环等等。而Less不支持。

### 5. 引用外部CSS文件

scss引用的外部文件命名必须以\_开头, 如下例所示:其中\_test1.scss、\_test2.scss、\_test3.scss文件分别设置的h1 h2 h3。文件名如果以下划线\_开头的话，Sass会认为该文件是一个引用文件，不会将其编译为css文件.

### 6.Sass和Less的工具库不同

Sass有工具库Compass, 简单说，Sass和Compass的关系有点像Javascript和jQuery的关系,Compass是Sass的工具库。在它的基础上，封装了一系列有用的模块和模板，补充强化了Sass的功能。

Less有UI组件库Bootstrap,Bootstrap是web前端开发中一个比较有名的前端UI组件库，Bootstrap的样式文件部分源码就是采用Less语法编写。

不管是Sass，还是Less，都可以视为一种基于CSS之上的高级语言，其目的是使得CSS开发更灵活和更强大，Sass的功能比Less强大,基本可以说是一种真正的编程语言了，Less则相对清晰明了,易于上手,对编译环境要求比较宽松。考虑到编译Sass要安装Ruby,而Ruby官网在国内访问不了,个人在实际开发中更倾向于选择Less。

**1.GIT是分布式的，SVN不是**

GIT跟SVN一样有自己的集中式版本库或服务器。但，GIT更倾向于被使用于分布式模式，也就是每个开发人员从中心版本库/服务器上chect out代码后会在自己的机器上克隆一个自己的版本库。可以这样说，如果你被困在一个不能连接网络的地方时，就像在飞机上，地下室，电梯里等，你仍然能够提交文件，查看历史版本记录，创建项目分支，等

同样，这种分布式的操作模式对于开源软件社区的开发来说也是个巨大的恩赐，你不必再像以前那样做出补丁包，通过email方式发送出去，你只需要创建一个分支，向项目团队发送一个推请求。这能让你的代码保持最新，而且不会在传输过程中丢失

**2.GIT把内容按元数据方式存储，而SVN是按文件**

所有的资源控制系统都是把文件的元信息隐藏在一个类似.svn,.cvs等的文件夹里。如果你把.git目录的体积大小跟.svn比较，你会发现它们差距很大。因为,.git目录是处于你的机器上的一个克隆版的版本库，它拥有中心版本库上所有的东西，例如标签，分支，版本记录等。

**3.GIT分支和SVN的分支不同**

分支在SVN中一点不特别，就是版本库中的另外的一个目录。如果你想知道是否合并了一个分支，你需要手工运行像这样的命令[*svn propget svn:mergeinfo*](http://jan.baresovi.cz/dr/en/subversion-mergeinfo)，来确认代码是否被合并

然而，处理GIT的分支却是相当的简单和有趣。你可以从同一个工作目录下快速的在几个分支间切换。你很容易发现未被合并的分支，你能简单而快捷的合并这些文件。

**4.GIT没有一个全局的版本号，而SVN有**

**5.GIT的内容完整性要优于SVN**

GIT的内容存储使用的是[SHA-1](http://en.wikipedia.org/wiki/SHA-1)哈希算法。这能确保代码内容的完整性，确保在遇到磁盘故障和网络问题时降低对版本库的破坏