一定要自信！底气十足！语调稍微俏皮活泼一点，不要像背书一样喔！

回答问题：www是什么、为什么、怎么样，即概念+优点+用法。其实最好是概念+原理+优点分析。

## 自我介绍

### 1分钟版本

面试官您好，我叫王慧兰，是来自浙江大学的研二在读硕士生，预计明年3月份毕业。本科就读于华中科技大学，期间先后获得过国家励志奖学金和国家奖学金，2018年本科毕业后保送至浙江大学读研。

读研期间我做过两个项目，一个是和北京301解放军总医院合作的“高原生命刻画与决策支持系统”，这个项目的背景是设计一个系统来记录高原士兵的生命体征及进行患病风险评估。作为主要负责人，我设计了前端UI并用Angular框架独立进行开发，使用SpringBoot搭建后台服务并用java编写大部分后台代码。最终系统实现了信息录入、统计展示及推理诊断的功能。第二个项目是将基于C/S架构的winform版决策支持软件重构为基于B/S架构的web应用，我的工作包括将原来基于C#的后台服务打包为web service工程，重写接口。然后设计新的web UI，并基于Jquery、HTML和CSS进行前端页面开发, 通过Ajax调用后端服务的Restful API，最终实现了页面动态效果与数据流动。

（面试官问起来再说选择前端的原因，见下）

### 3-5分钟版本

面试官您好，我叫王慧兰，是来自浙江大学的研二在读硕士生，研究方向是医学信息学和医疗软件工程，预计明年3月份毕业。本科就读于华中科技大学，本科期间先后获得过专业top5的国家励志奖学金和top1国家奖学金，2018年本科毕业后保送至浙江大学读研。

在读研期间，我做过两个项目。一个是和北京301解放军总医院合作的“高原生命刻画与决策支持系统”，这个项目的背景是设计一个系统来记录高原士兵的生命体征，比如在高原上表现出的各种症状，并根据这些信息对他们是否会患高原病进行风险评估以及给出干预建议。作为主要负责人，我先通过文献调研和请教医生确定系统需要录入的信息项，以及确定高原病诊断标准和治疗建议。然后明确系统的模块和功能，设计前端UI并用Angular框架独立进行开发，使用SpringBoot框架搭建后台服务并用java编写大部分后台代码。最终系统实现了信息录入、统计展示及推理诊断的功能，并交由301医院推广。

另一个项目是更早一些时候的一个研究生入学技能培训项目，要求将基于C/S架构的winform版决策支持软件重构为基于B/S架构的web应用。这个winfrom版应用是实验室师兄前几年开发的。我的工作主要包括先将原来基于C#的后台服务打包为web service工程，重写接口。然后设计新的web UI，并基于Jquery、HTML和CSS进行前端页面开发, 通过Ajax调用后端服务的Restful API，实现页面动态效果与数据流动。

在做这两个项目的过程中，我发现自己对前端开发有比较浓厚的兴趣和探索欲，看到自己写的代码变成网页显示在浏览器中时，会有较大的成就感。并且我会思考怎样让代码更加简洁规范和高效。使交互性更好，怎样更美观等等。这样的兴趣和热情让我决定以后想成为一名前端开发工程师。介绍完毕，请您指教。

## 非技术问题：

### 美团二面/三面

### 为什么选前端？（项目经历里既有前端又有后端）

×：喜欢思考怎样使交互性更好，使页面更美观。**这是UI设计师的工作**！

√：自己实现了一些视觉上可见的页面和交互，会比较有成就感，感觉很有意思。当然也具有挑战性，有时候遇到一些自己不太会实现的交互或者样式，我就会想办法去实现，比如在网上搜，看官方文档或者请教别人，在这个过程中，整个人的状态非常专注投入，感觉很充实，不觉得辛苦或疲惫，我很喜欢这样的状态。

**TODO：**还要再深入谈一下对前端的思考！

### 怎么学习前端的？（进一步）有没有系统地看过什么书？（推荐了高级程序设计3）

刚开始在慕课网上看了一些基础的关于js，html和css的教学视频，然后就开始动手做项目了，项目中遇到不会的问题或者发现了一些模棱两可的知识点，就会搜一些相关的博客来进一步学习。后面又看了《JavaScript dom编程艺术》这本书，在网易云课堂上跟了一个梳理ES6的教学视频，然后现在在看《HTML5与CSS3权威指南》，准备再看一下**javaScript高级程序设计3（面试官似乎都很认可这本书！）**。

### 你是怎么设计高原这个系统的？能不能具体说一下思路？

首先明确这个系统的大概需求。然后列举这个系统要实现什么功能，有几个模块，那调研后就确定下来主要有病人信息录入（包括增删改查），统计分析和推理诊断这几个大的功能模块，然后参考了一些网站的设计设计了系统的UI，弄清楚各个页面的内容和页面之间的跳转，然后就是写代码实现 了。

### js和其他语言（如java）的区别

1. java是编译型语言，需要先编译再运行，javascript是脚本语言，在程序运行过程中被逐行地解释。
2. java是静态语言，定义变量时需要指明类型。javascript是动态语言，变量本身类型不固定。
3. java面向对象，javaScript则基于原型，在es6之前都没有class关键字，用函数来模拟类实现。
4. java主要运行在服务端，javaScript运行在客户端浏览器中。

### 做项目的过程中有没有遇到什么难点？

其实刚开始学习的时候觉得都挺难的，有点难理解Angular这个框架，感觉跟用原生js开发差别太大了。然后通过不断的学习逐渐上手了，令我印象比较深的是：**在父组件中设置子组件中的表单的值。**父组件是一个病人列表，每个病人都有一个编辑信息的按钮，点击按钮会弹出一个模态框，显示这个病人的基本信息。我是将这个模态框以子组件的形式放在父组件的，一开始不知道要怎样把从后台传来的数据填充到子组件的模态框的表单中，而且重新选别的病人能跟着动态更新。在网上搜了半天也没搜到，我发现网上关于angular的讨论有点少，不像react/vue一搜出来就一大堆，后来仔细找了一下Angular官方文档，在生命周期那块发现通过带@ViewChild装饰器的属性来访问子视图。

职业规划？

现在投了几家公司？有没有offer？

为啥没有投阿里、腾讯这些大厂？

## Angualr框架

### 怎么如何理解Angular？Angular的核心概念/模块？

Angular 是一个用 HTML 和 TypeScript 搭建前端的平台与框架。依据我的项目体验和参考官方文档，我认为Angular主要有以下这8个核心概念：模块、组件、元数据、模板、数据绑定、指令、服务以及依赖注入。Augular的核心是模块化和组件化，一个模块就是一个容器，包含多个组件和其它文件，集中实现某一个工作流或者一组紧密相关的功能。组件是Angular的基本构造块，组件（类）和模板由元数据关联起来，共同定义了Angular的视图view。模板又是普通HTML和Augular指令及数据绑定标记的组合，通过数据绑定可以轻松地在视图中显示数据、监听用户的输入并进行相关的处理和视图更新，通过指令则可以轻松的添加、移除DOM以及改变HTML元素外观。组件聚焦于用户体验，主要提供用于数据绑定的属性和方法，其他处理任务比如从服务器获取数据等则是交由服务service来完成的，通过依赖注入机制将服务注入到组件，让组件使用该服务，服务还可以注入到其他需要的组件中以供访问。Angular通过将组件和服务区分开来，提高了模块化和复用性，使组件类更加简洁、高效。

### 说一说Angular的生命周期

首先生命周期这个概念是针对Angualr中的组件或指令来说的，一个生命周期指的是组件/指令从被创建、到更新，再到最后被销毁这样的一个过程。Angular 提供了生命周期钩子，使我们可以在组件生命中的这些特殊时刻，进行一些处理。具体来说，每个生命周期钩子都是一个接口，每个接口对应唯一一个钩子方法，我们根据需要自行选择在组件中定义哪些钩子方法，当我们使用构造函数新建一个组件或指令后，Augualr就会按照一定的顺序在特定的时刻调用这些钩子方法。

Angular有8种生命周期钩子方法，按照在每个生命周期中调用的先后顺序依次为：

第一个ngOnChanges( )：当 Angular（重新）设置数据绑定输入属性时响应。按我的理解，就是子组件的@input输入属性发生变化时调用，而且每次发生变化都会调用（该方法接受当前和上一属性值的 SimpleChanges 对象。）

第二个ngOnInit( )：这是一个最常用的钩子方法，在 Angular 第一次显示数据绑定和设置输入属性之后响应，用来初始化指令/组件，所以一般在这里放置初始化逻辑。在第一轮 ngOnChanges() 完成之后调用，**只调用一次**。

第三个ngDoCheck()：用来检测那些被 Angular 忽略的更改，这个钩子方法要谨慎使用，因为它被调用的太频繁了，页面上任何地方发生更改都会被捕捉到（比如仅仅把鼠标移到其它 <input> 中就会触发一次调用。 只有相对较少的调用才是由于对相关数据的修改而触发的。）

第四个和第五个分别是ngAfterContentInit()和ngAfterContentChecked()，是和外部内容投影有关的，ngAfterContentInit()在 Angular 把**外部内容投影进视图之后**调用，第一次 ngDoCheck() 之后调用，只调用一次。ngAfterContentChecked()则当每次被投影内容中的值发生变化时响应，和ngDoCheck一样，响应很频繁。

第六个和第七个分别是ngAfterViewInit()和ngAfterViewChecked()，这两个钩子方法和前面的ngAfterContentInit()和ngAfterContentChecked()很相似，只是它们针对的是视图及其子视图，ngAfterViewInit()当 Angular **初始化完组件视图及其子视图之后调用**，只调用一次。而ngAfterViewChecked()是每当视图和子视图中有数据变更就被调用。

最后一个是ngOnDestroy(),在 Angular 销毁指令/组件之前调用, 释放那些不会被垃圾收集器自动回收的各类资源。（取消那些对可观察对象和 DOM 事件的订阅。停止定时器。注销该指令曾注册到全局服务或应用级服务中的各种回调函数。 如果不这么做，就会有导致内存泄露的风险。）

### Augular中的数据绑定

数据绑定是一种用来协调模板和组件中各个部分的内容的机制，往模板 HTML 中添加绑定标记可以告诉 Angular 该如何去找到这些内容。数据绑定的目标是 DOM 中的对象。根据绑定类型，该目标可以是HTML元素、组件或指令的属性名、也可以事件名

1. 属性绑定：是指数据从组件流入模板html，从而将在组件类中定义的数据显示在视图中。
2. 事件绑定：数据从模板流入组件，通过调用组件中定义的方法来响应用户的输入
3. 双向绑定[(ngModel)],顾名思义就是属性绑定和事件绑定的组合，主要用于表单控件中，数据属性值通过属性绑定从组件流到输入框。用户的修改通过事件绑定流回组件，把属性值设置为最新的值。

如果没有数据绑定，就要自己负责把数据值推送到 HTML 控件中，并把来自用户的响应转换成动作和对值的更新。手动写这种数据推拉逻辑会很枯燥、容易出错，难以阅读。有了数据绑定，应用就会更易于编写、阅读和维护。我们就只需声明数据源和目标 HTML 元素之间的绑定关系就可以了，框架会完成其余的工作。

### Angular中的指令？

从实现上来说，指令就是一个带有 @Directive() 装饰器的类。Angular中有三种类型的指令，组件、结构型指令和属性型指令。组件其实就是一种带有模板的指令，但因为组件在angular中非常重要和特殊，因此Angular 专门定义了 @Component() 装饰器来表示组件类。

结构型指令通过添加和移除 DOM 元素改变 DOM 布局。比如最常用的有Angular的内置结构型指令\*ngFor和\*ngIf，我们把结构型指令应用到某个宿主元素上，然后它就可以改变宿主元素及其子元素的DOM布局。比如ngFor指令会重复渲染宿主元素，而ngIf则根据赋给它的布尔值让宿主元素显示或消失。（星号是一个用来简化更复杂语法的“语法糖”。 从内部实现来说，Angular 会把星号（\*）语法解开成 <ng-template>并用它来包裹宿主元素）

属性型指令则用来改变一个元素的外观或行为，通常会作为HTML属性的名称而应用在元素上。比如常用的有angular的内置属性型指令ngClass和ngStyle，通过绑定到NgClass，可以同时添加或移除多个css类，而NgStyle绑定可以同时设置多个内联样式。

除了angular内置指令，我们也可以自定义指令。自定义指令和创建组件差不多，只是导入 Directive 装饰器（而不再是 Component），@Directive 装饰器的配置属性中指定了该指令的 CSS 属性型选择器 ，以便在模板中标识出这个指令该应用于哪个元素。结构型指令需要导入符号 Input、TemplateRef 和 ViewContainerRef，使用TemplateRef取得 <ng-template> 的内容，并通过ViewContainerRef来访问这个视图容器。属性型指令一般需要导入ElementRef，通常在指令的构造函数中使用 ElementRef 来注入宿主 DOM 元素的引用，ElementRef 通过其 nativeElement 属性可以直接访问宿主 DOM 元素。

### vue、angular等相对于 jQuery在开发上有什么优点?（Angular比较好的功能或者优点是什么？（携程一面））

ngFor指令在导航中的用处，还有渲染有很多输入的表单，非常方便!

### 有没有去了解一些angular的原理的一些东西？比如angular的数据绑定是什么机制？

### 对 MVC、MVVM的理解（美团）