**什么是前端开发者**

[互联网平台](https://en.wikipedia.org/wiki/Open_Web_Platform)[NativeScript](https://www.nativescript.org/)1. 前端开发者使用互联网技术（比如HTML、CSS、DOM和JavaScript）建造并开发网站和应用，这些网站和应用会运行在上，或者作为无网络环境的输入应用（比如 ）。

访问您网站的访问者可以看到，点击或用于输入或检索信息的方式是前端开发人员创建客户端软件的工作，从而使网站的设计变得生动。脚本由浏览器下载，处理，然后与服务器分开运行。

2. 前端和后端分工的三种模式

 在Web开发中，前端攻城狮和后端攻城狮是不同的物种，一个追求任何场景下都美丽动人，一个追求巨大压力下举重若轻。但两者又必须密切分工合作，才能使得项目顺利进行。分工的核心在于在哪里渲染页面。不同的渲染位置决定了不同分工模式。

      渲染是把数据填充进模板，按模板定制的样式把数据展示出来。如下图所示知乎的例子，图左上方的模板定制了我们看到的样子，其中？表示没有数据，这时将一个用户的数据（图左下方）填充进模板，便得到了我们看到的页面(图右边)。这个过程就是渲染。一共有三种渲染的方式。



2.1 在服务器端渲染

      浏览器发送请求到服务器端，服务器端处理请求并准备好数据，然后将数据写进前端编写的模板中，从而生成html文件，将生成的html发回给浏览器。这样浏览器上就显示页面了。

      <table class="table">

      <tr>

            <td>项目名称</td>

            <td>组长姓名</td>

      </tr>

      <% for(var i = 0; i < project.length; i ++){ %><% } %>

      <tr>

            <td><%= project[i]['project'] %></td>

            <td><%= project[i]['leadername'] %></td>

      </tr >

      <% } %>

      </table>

      上面是一个模板。这份模板是一个html文件，其中带有数据绑定命令"<td><%= project[i]['project'] %></td>"。当后端程序整理好数据，在服务器端将数据填充模板，渲染出页面。如下代码所示，app.render(模板,project)语句的意思是，在服务器端将projectdata填充进模板生成页面，并将其发送给浏览器。

      app.get("/show",function(req,res){

            projectdata = preparedata();

            render(模板,projectdata);

      })

      这个模式有一个问题——不能实现部分更新。即使用户点了一个按钮，产生了很细微的数据变动，也需要后端重新渲染整个页面再将页面发往浏览器端。如果页面存在大量的静态的部分，这种方式无疑不是高效的。

      同时，前端工程师们需要用模板定义展现形式，后端工程师们需要用模板输出数据。久而久之，模板就会越来越复杂，越来越不可维护。

2. 2 在浏览器端渲染

      现在一个趋势是渲染移动到浏览器完成。浏览器端发送请求后从服务器端接受到了模板和 J S代码。浏览器执行接受到的 JS 代码，JS 代码会从服务器请求数据，并将数据填充到模板中。下面的代码在页面加载完之后从接口 /online/projectlang 获取项目语言的数据，并将其写入html页面。

      $(document).ready(function(){

      $.ajax({

         url:"/online/projectlang",

         type:'get',

         dataType:'json',

         success:function(data){

            var list = '';

            for(int i = 0 ; i < data.length; i ++){

               var item = "<li><a href=\'#\'>"

               + data[i] + "</a></li>";

               list += item;

            }

            $("#projectlang").append(list);

         },

         ;error:function(xhr, status, error){

            console.log("projecthot ajax "+ error);

         },

      });

      })

      利用运行在浏览器端的Javascript语言，前端工程师能够从后端服务器获取数据，进而按照业务逻辑渲染页面。这时候后端工程师只需要开发稳定的 API 提供数据就可以了。这种模式虽然依然是B/S模式，但开发的场景却和C/S模式比较相近。在浏览器端渲染的好处在于前端完全控制了模板，后端只需要开发相应的 API, 分工比较明确。并且支持部分页面更新。同时同一套后端服务可以同时支持不同的展示模式，比如同一套后端服务还可以支持移动开发。

      当然啦，浏览器渲染也存在一些问题。其中最大的问题是对 SEO 不友好。搜索引擎的爬虫程序必须像浏览器一样执行 JS 代码才能获得页面的内容，从而提高了爬虫爬取页面的难度。

   3.2大前端模式

      借助神器Node.js，前端工程师终于把磨爪伸进服务器了。Node.js是一个来自老毛子的高性能异步服务器。如果只是一个服务器，Node.js并不出奇。一般服务器需要提供一种编程语言的runtime，方便开发者进行开发。Node.js因为异步的关系选择了异步性能很好的Javascript，就是前端工程师经常在页面上写的那个Javascript。这时前端工程师们一看，呀，这玩意我会呀。因此利用Node.js，前端工程师不再局限在浏览器，可以在服务器写Javascript代码了。这时前端工程师可以按需要，选择在浏览器端或者服务器端完成渲染。这个模式我们可以称之为大前端模式。

**前端职位分类**

下面是各种前端职位的介绍。通常来说，或者最普遍来说，这个职位一般叫做“前端开发”或者“前端工程师”。基本上，所有包括“前端”、“客户端”、“网站UI”、“HTML”、“CSS”或者“JavaScript”的职位都意味着这个人对HTML、CSS、DOM和JavaScript有某种程度的认识与实践。

**前端开发/工程师**（也就是前端网页开发/工程师或者客户端开发/工程师或者前端软件开发/工程师，所有带有Front-end, Web, Software, Develper, Engineer的排列组合）

这个职位表示某人拥有某种程度的HTML、CSS、DOM以及JavaScript技能，并且能够将其应用于互联网平台上。

**CSS/HTML 开发**

这个职位表示这个开发者者掌握 HTML 和 CSS 技巧，不包括 JavaScript 和应用实践。

**前端JavaScript（或者应用）开发**

当职位标题中含有“JavaScript应用”的时候，这表示这个开发者应该是一个高级JavaScript开发者，并拥有可以编写高级程序、软件或应用的技能（比如，拥有扎实的前端应用开发经验）。

**前端页面设计师**

当职位标题中含有“设计师”，这意味着这个人应该拥有前端技巧，并且同样精于设计。

**网络/前端用户界面（UI）开发/工程师**

当职位标题中含有“界面”或者“UI”的时候，这意味着这个开发者除了掌握普通的前端技巧之外，还应该掌设计、交互设计以及线框图的技能。

**移动/平板前端开发**

当职位标题中含有“移动”、“平板”的时候，这意味着这个开发者拥有开发移动端或者平板端的经验（包括完全本地化的或者网页平台）。

**前端SEO专家**

当职位标题中含有“SEO”的时候，这意味着这个开发者对SEO策略具有大量经验，可以给予或者构建。

**前端易用性专家**

当职位标题中含有“易用性”的时候，这意味着这个开发者拥有大量关于易用性需求和标准的编码经验。

**前端DevOps**

当职位中包含“DevOps”的时候，这意味着这个开发者拥有大量软件开发方面的经验，强调沟通、协作、集成、部署、自动化以及测试。

**前端测试/QA**

当职位中含有“测试”或者“QA”的时候，这意味着这个开发者在测试和评测软件方面有大量经验，包括单元测试、功能测试、使用者测试、A/B测试。

**前端开发所需的web技术**

超文本标记语言（Hyper Text Markup Language，也就是HTML）

层叠样式表（Cascading Style Sheets，也就是CSS）

文档对象模型（Document Object Model，也就是DOM）

JavaScript程序语言（包括ECMAScript 6, ES6, JavaScript 2015）

网络API（也就是 HTML5 和他的小伙伴们，或者浏览器API）

超文本传输协议（Hypertext Transfer Protocol，也就是HTTP）

统一资源定位符（Uniform Resource Locator's，也就是URL）

JavaScript对象表示法（JavaScript Object Notation，也就是JSON）

网络内容易用性指南（Web Content Accessibility Guidelines，WCAG）和易用富网络应用（Accessible Rich Internet Applications，ARIA）

**前端面试问题**

你可能会被问到的问题：

[Front-end Job Interview Questions](http://h5bp.github.io/Front-end-Developer-Interview-Questions/)

[Interview Questions for front-end-Developer](http://thatjsdude.com/interview/index.html)

[10 Interview Questions Every JavaScript Developer Should Know](https://medium.com/javascript-scene/10-interview-questions-every-javascript-developer-should-know-6fa6bdf5ad95)

你可以问的问题：

[An open source list of developer questions to](https://github.com/ChiperSoft/InterviewThis)

**前端求职**

现今有大量技术求职网站。下面的精简列表是目前与寻找特定前端职位相关性最高的网站。

[frontenddeveloperjob.com](http://frontenddeveloperjob.com/)

[authenticjobs.com](https://authenticjobs.com/#category=4)

[weworkremotely.com](https://weworkremotely.com/)

[jobs.github.com](https://jobs.github.com/)

[careers.stackoverflow.com](http://careers.stackoverflow.com/jobs?searchTerm=front-end)

[angularjobs.com](http://angularjobs.com/)

[jobs.emberjs.com](http://jobs.emberjs.com/)

[jobs.jsninja.com](http://jobs.jsninja.com/)

[css-tricks.com/jobs](https://css-tricks.com/jobs/)

[glassdoor.com](http://www.glassdoor.com/Job/front-end-developer-jobs-SRCH_KO0,19.htm?jobType=all)

**前端是怎样炼成的**

1. 前端的学习是自我学习的过程(computer science的学历只能说明有基础)。

2. 前端的开发不是视觉的交互。

[原生网络应用（也就是SPA）](https://blog.andyet.com/2015/01/22/native-web-apps)3.学习，网络是如何工作的。确定你知道“什么”是域名、DNA、URL、HTTP、网络、浏览器、服务器/主机、数据库、JSON、数据API、HTML、CSS、DOM、JavaScript以及它们都“在哪”。目标是确定你大体了解这些东西是如何协同工作，以及每个部分具体都做些什么。关注高度概括的前端架构。从简单的网页开始学习。

学习HTML、CSS、易用性和SEO。

学习UI设计模式、交互设计、用户体验设计和可用性的基础。

学习编程基础

学习JavaScript

学习JSON和数据API

学习CLI/命令行

学习软件工程实践（应用设计/架构、模板、Git、测试、模拟、自动化、代码质量、开发方法）。

弄一些适合自己大脑的自用、定制化工具盒。

学习Node.js