**框架**

**Vue**

1. vue的特点
2. jQuery和vue的区别
3. Vue的生命周期
4. vue跨组件通信实现
5. vue怎么样实现数组绑定
6. vue双向绑定原理，vue-loader做了什么
7. vue的props和slot的使用区别
8. vuex的状态管理的原理是什么?
9. vue diff算法
10. vue-router不能解决情况和边界情况

**Webpack**

1. webpack工程构建工具怎么样用
2. webpack的原理
3. webpack的插件大致流程
4. webpack项目太大了怎么办
5. webpack的plugin和loader的区别

**算法**

1. 说下快排完整性
2. 数组去重
3. 快排和冒泡原理
4. 一个有序的数组如何进行查找操作
5. 手写一个快速排序
6. 数组常用的方法有哪些
7. 尾递归问题
8. 算法题：二叉树层序遍历
9. hashmap原理
10. 如何去除数组中的重复的项？
11. 算法：两个排序好的数组，怎么样找他的中位数

**HTML和CSS样式**

1. 如何理解html语义化

|  |
| --- |
| HTML语义化是为了解决div soup的问题，让HTML更具有可视化。通过<header>,<nav>,<main>,<article>,<footer>的应用，可以在接手别人的代码的时候，对结构一目了然。减少沟通成本。 |

1. 浏览器如何处理未知的的tab
2. 为什么css样式初始化，目的是为了什么？

|  |
| --- |
| 浏览器css会有一些default属性，比如margin，padding不为0，box-sizing是content-box属性之类的。为了和UI设计在统一的测量标准下，将这些属性置0，在构建视图时工作效率会提高。 |

1. css弹性布局，那些地方用到过
2. position属性有哪些值，分别有什么含义
3. 清除浮动的方法
4. 常见布局的方法
5. 实现一个两边宽度固定中间自适应的三列布局，圣杯布局，双飞燕
6. flex布局有没有了解
7. 实现三栏布局，中间自适应有几种方法
8. CSS动画会吗，怎么样用CSS实现一个loading效果
9. 如何处理CSS兼容问题
10. margin坍塌
11. BFC原理
12. 写一下不知道宽高元素垂直水平居中方法
13. 如何获取元素的父节点和兄弟节点
14. 写一下节点的增删改
15. 如何画一个三角形
16. CSS中对溢出的处理
17. CSS选择器有哪些，优先级呢？
18. boostrap的底层原理？
19. flex弹性布局裂解，移动端适配方案有哪些？
20. float和position的区别
21. 计算器
22. 盒子模？border-box和content-box？
23. css伪元素有哪些
24. css性能？css动画优化？
25. css性能优化，就动画效果，如何从js，cs角度减少回流？
26. 盒子米线
27. rem，px，em的区别
28. HTML的黑钻模型有哪些构成的，盒子模型有哪些，默认是哪些
29. flex弹性布局
30. block元素和inline元素的区别
31. position有哪些，特性
32. css选择器的优先级

**Javascript**

**Js的原理**

1. js的继承？实现方法？
2. js的this理解， 如何改变this的指向

|  |
| --- |
| This这个参数到底代表谁，实际上是代码执行时动态形成的。要判断this的指向，需要明确的是含有this的函数到底在哪里进行了调用。This即指代的是进行调用的Object，将它作为上下文传入正在执行的函数。  ///////////////////////////////////////////////////////////////////////////////////////////////////  更改this的指向：隐性更改的话变换调用的Object即可，显性更改的话使用call、Apply、Bind函数进行再次绑定。  ///////////////////////////////////////////////////////////////////////////////////////////////////  这道题目的潜台词也是在问怎么解决this的绑定丢失的问题。回调函数经常造成this的绑定丢失，this是在执行的时候生成的指向。传进给回调函数里一个this，它没有再执行调用了（函数只执行一次），指向是undefined，要使用bind绑定住对象。    经典的this绑定丢失案例，解决方案：obj.foo.bind(obj); |

1. js的事件机制
2. JS基本数据类型
3. 内存泄露的排除定位和解决方法
4. 垃圾回收机制
5. 请简单说明什么是事件冒泡和事件捕获以及事件委托
6. 闭包

|  |
| --- |
| 闭包的概念是：闭包记住了声明scope下的变量，在别的时间空间下，调用函数时，依然可以读取到这些变量。    这是闭包的经典案例，这里面提出来的问题就是outer执行结束之后，函数空间消亡了，inner的n为什么还有值？  闭包在我的理解里，有着很重的时间空间属性。闭包有点像inner和outer之间的虫洞，保持着inner保持的一个通道可以访问到outer中的变量n。由于“虫洞”的存在，outer空间不会消亡，直到“虫洞”访问需求消失。 |

1. let和var的区别，let的产生背景？

|  |
| --- |
| 这两个关键词的最关键的区别是scope，var是认定的scope不包含for，if等这类函数块的，let把for，iF等函数块认定为一个scope。如果说有一些需要在函数块中记住变量属性的表达式，ES5下写，需要加上花括号，这样子嵌套的层级会很多。我猜测是为了代码整洁的原则，ES6的时候就发明了let来简化代码。 |

1. var的变量提升底层原理是什么？

|  |
| --- |
| 这是由JS编译造成的。一段代码的执行，实际上分为两个步骤，第一步骤对声明进行编译：这一步为var或function分配地址内存。第二阶段执行阶段：这一步就是正常的赋值或者运算等执行过程。  例如var a = 2; 编译器先不会管a = 2这步骤，它第一步要做的是把var a进行编译，而执行留在原地。声明在编译角度上被移动到了作用域的顶端。  这就是提升的底层原理。 |

1. 模块化思想

|  |
| --- |
| 大致从什么是模块化？为什么要使用模块化？以及怎么写一段模块代码来梳理回答思路。  模块化思想即最小暴露原则。隐藏掉别人不需要知道的代码细节，仅提供需要使用的API。保护代码的安全性和可维护性。  书写一段模块的要点有二：1、要使用闭包原理。2、函数至少被调用执行过一次。  方法一：工厂模式    方法二：单例模式（IIFE） |

1. JS的回收机制说一下
2. 事件委托理解，原理，好处，应用场景
3. 请简述一下js原型链

|  |
| --- |
| 一般描述类，多为JAVA和C++的类概念。这种类实际上是一种拷贝行为。但是JS的类是基于JS原型链原理生成的。Js生成的实例，与父类之间形成一条链接，如果生成类中没有的属性，通过链接回溯到父类中寻找，这种行为一直可以回溯到Object.Prototype为止。比起拷贝的类型，这种原型链形式，实际上更加节省内存。另外由于javasctipt的面向对象oo，实际上是oloo。 |

1. 在原型链上Object再往上是什么？

|  |
| --- |
| Object.prototype是原型链的终点了。 |

1. new生成了一个对象的过程（核心return this）

|  |
| --- |
| New关键字生成对象分为4个步骤：1、生成空白对象，2、将空白对象作为上下文传入父类。3、将父类中的this与新生成的对象进行绑定。4、父类返回值赋值给新的对象。    New做的总做就是line 2 和 line 6的工作，简约了代码。 |

1. new和Object.create的区别

|  |
| --- |
| New和Object.create基本上完成的功能很像，创建一个新的空对象，链接到另一个对象。但new还有点不同，new通过构造函数创建对象。Object.create方法只创建了原型链（访问父类的渠道）。代码实例：    Amy.constructor指向的不是Van，而是Car，因为Van不包含constructor函数。 |

1. \_proto\_ 和prototype分别是什么？哪种情况下\_\_proto\_\_和prototype的指向是同一个？

|  |
| --- |
| 这道题目问的有些奇怪，大体上考察的问题：dunder和protype的指向。  直接上例子：    解析原型链的布线，需要明白的是函数的模型    子类对象通过\_\_protorype\_\_指向父类的对象；函数通过Prototype指向同级对象 |

1. 原型链断裂了以后的结果是什么
2. 浏览器是如何实现通过你的代码去找到指定的元素的
3. 手写一下深拷贝，答案提示：JSON.parse(JSON.stringify(obj)) 用JSON实现深拷贝
4. 浏览器的缓存
5. 前端性能理解，优化有哪些
6. JS如何计算浏览器的渲染时间的

**Js的语法**

1. preventDefault和stopPropagation的区别
2. 遍历数组处理用for循环还有什么方法吗？
3. instanceOf和constructor的区别
4. typeof array null undefined NaN分别是什么
5. 把undefined和null转成Number分别是什么
6. 说说bind，apply，call的区别以及bind的实现
7. jsonp的原理是什么，处理script标签还可以通过什么实现？（静态资源标签）
8. 你说到typeof，能不能加一个限制条件typeof只能判断是object，可以判断一下是否拥有数组的方法
9. 如何判断是否为数组？（instanceOf和constructor可以，但是有原型链断裂的风险，Object.toString.call()最稳定）
10. == 和 === 的区别，，typeof null的结果是什么
11. event loop讲讲
12. settimeout的机制

**语言的应用题**

1. 写一个API，实现jQuery的$(selector)选择器，要求兼容IE6
2. JS的如何实现倒计时，为什么不准，校正方式
3. JS如何获得用户来源（navigator.userAgent）
4. 如果让你实现一个promise怎么样实现
5. 用JS模拟一个双向链表

**AJAX**

1. ajax是同步还是异步，怎么样实现同步
2. ajax是同步还是异步，怎么样实现同步
3. ajax实现过程
4. get和post的请求区别
5. 发送请求有哪些？
6. Ajax如何实现的
7. Ajax返回204算是成功吗

**ES6**

1. ES6用过吗，新增了那些东西，你用到过什么
2. const和let的区别，可以改变const定义对象某个属性吗
3. 箭头函数，箭头函数的特点
4. es6有了解吗，请简单说一下promise机制，异步的承诺机制，顺势说一下解决回调地狱的问题
5. 同步和异步的执行顺序
6. ES6新特性说说
7. ES6的代理是什么
8. 箭头函数和ES5和this的指向区别讲讲
9. es6和es5的区别
10. es6模块新特性
11. es6的变量定义和es5的区别

**实战**

1. 你项目里用到第三方登录涉及的oAuth(JWT)协议的实现原理，以及你本地的实现原理，第三方登录怎么样保证安全性。
2. 工作中做的项目有什么亮点
3. cookie和webstrage的区别以及cookie怎么样使用？原生cookie怎么样封装
4. cookie有什么限制
5. 遇到过兼容性的问题吗，要如何处理
6. 项目中使用过构建工具吗
7. 代码规范
8. 现在有一大段文字，如何在页面中设置一个窗口滚动播出这段文件（轮播），轮播图如何解决卡顿问题，有手写过轮播图你，原理是什么
9. 了解web移动开发吗，移动端适配方案有哪些

**网络**

1. JS实现跨域
2. 什么情况算是跨域，跨域实现和原理？如何解决跨域问题？
3. http状态码
4. 为什么浏览器会产生同源策略
5. 三次握手
6. 对跨域了解吗。jsonp的限制
7. 浏览器那些地方用到了异步
8. http状态码301 302的区别，304是啥
9. 从输入一个url到浏览器页面展示都经历了哪些过程
10. 缓存机制，协商协议
11. https的请求过程









 





 编程题：给出一个字符串(“obj.a”)，返回对象属性obj.a，类似eval的效果

 画布濡染有了解吗





 深拷贝

 如果弹出的菜单位置过于贴近边框，如何调整这个元素的位置

 计算50个人至少有2个生日相同的概率

 一个升序数组，求两个元素的和为一个指定数



 如何在浏览器端和原生端的代码复用，讲到weex又简单讲了它的原理



 前端性能优化问题



 get和post的区别

 操作系统进程和线程的区别

 线程的哪些资源共享，哪些资源不共享



 promise的状态有哪些

 实现一个两列等高布局，思路

 如何让一个元素消失

 重排和重绘

 浏览器缓存机制

 强缓存，协商缓存什么时候用哪个





 前端优化

 实现一个ajax，兼容

 如果有这样的场景：一个模块A作为输出，BCD等扩展模块可以在A做更改后展示A的原来内容或者驾驶CSS后的内容，想想思路







 页面缓存

 页面性能优化



 正则如何将一个数千分化表示

 js设计模式知道哪些，单例详细说

 函数式编程-柯里化

 JS事件流

 排序算法有哪些，时间复杂度，选择排序怎么样搞

 数据结构有哪些，红黑树和二叉树的区别，二叉搜索树与二叉平衡树

 树数据多少非常多怎么样办（懒加载）

 页面优化方法

 设计模式

 递归

 js事件流



 promise如何从then转为catch的

 promise函数



 浏览器内存泄露，闭包内存泄露如何解决

 怎么样让页面加载更快

 兼容过IE的方法

 缓存

 会pc换还是移动端

 观察者模式如何实现



 es6的特性以及这些特性如何实现的

 事件冒泡，事件捕获，事件委托的原理，如何实现委托，事件，委托的有点是什么，事件监听

 输入URL浏览器是如何工作的

 requirejs组件化



 谈谈js设计模式

 如何实现订阅者发者模式

 MVVM实现原理

 详细说明解决跨域的方式

 HTTP状态码

 重排重绘

 谈谈JS的异步机制

 项目中是是如何优化页面的

 如何实现移动端布局，适配方案

 call apply bind的区别

 深拷贝的实现

 jQuery原理，平时用jQuery都做过什么

 用过webpack吗，谈一下webpack打包

 给了一个settimeout代码输出的顺序

 流式布局

 如何获取当前日期

 打开连接到网页呈现的流程

 HTTP请求头，响应头里面有哪些

 异步加载JS

 JS原型

 URL长度限制

 缓存

 浏览器兼容

 webpack css兼容



 reflow和repaint



 settimeout异步



transition生命周期

 线程和进程的区别

 什么是异步

 promise是如何解析异步操作的

 如果给a,b,c,d四个时间，执行时间分别为1，2，3，4，怎么样做才能在abc都在执行完后再执行d，除了使用promise还能使用什么方法

 函数节流和函数防抖知道吗，区别是什么

 强缓存和协商缓存知道吗

 排序的几种方法知道吗，是如何实现的

 栈和队列有什么区别，具体的应用场景



  给你一个DOM元素，用CSS的方式让他呈现两个的想过，只有一个DOM元素

 一个数组只有1和2，排序，1在前面，2在后面





 一个数组，有很多数字存在2次，只有一个数字寸一次，怎么样找出这个数字

 关于iframe内部和外部变量的读取是如何的

 前端防连击throttle和debounce

 ES6 generator async/await了解吗

* 如何反转链表

 二叉搜索树的原理

 给定两个文本文件，找出他们中相同的行都有哪些

 对JS单线程的理解

 页面共享数据的方法有哪些

 用户页面打开很慢，有哪些优化方式

 js如何实现一个栈

 说说操作系统的内存管理

 浏览器如何渲染页面的

 你是如何理解HTML语义化的



 清除浮动的所有方法

 基本数据类型

 typeof去判断数据类型返回值有哪些

 说说事件代理利用的是什么原理

 阻止冒泡的函数是什么

 cookie有什么特征

 加入访问A.com存进了一个cookie，在另外一个页面用ajax向A的域名发请求会携带cookie吗

 cookie的其他解决方案

 localStorage存储数据格式是什么

 怎么样把一个数组存进localSorage

 storage有哪些存储方法

 假设两台电脑之间同步画板如何实现

 promise的两个方法，具体实现

 es6用的多的有哪些

 箭头函数

 如果一个页面做辛夷花，从哪些方面考察，从哪些地方优化

 假设一个object A里面的值n为1，怎么样知道n改变了，有事件绑定吗

 给数字加千位分隔符

 webkit内核渲染页面过程

 简单介绍一下backbone

 typescript和es6的区别

 跨域方式



 预约系统如何解决高并发的问题

 简单说一下ICP UDP

 如何实现文件上传的功能

 jsonp如何实现，过程

localstorage和cookie的区别，如何设置cookie

 表单提交和ajax的区别

 symbol

 事件监听

 常见的请求方式

 情景题：给你一个ul列表，找到点击的li对象，把点击到的对象的字符串翻转，讲出思路并且写出代码

 只能用获取到对象的属性

 你怎么样维持用户登录状态的