Vue面试题

生命周期函数面试题

1.什么是 vue 生命周期
2.vue生命周期的作用是什么
3.第一次页面加载会触发哪几个钩子
4.简述每个周期具体适合哪些场景
5.created和mounted的区别
6.vue获取数据在哪个周期函数
7.请详细说下你对vue生命周期的理解?

vue路由面试题

1.mvvm 框架是什么?
2.vue-router 是什么?它有哪些组件
3.active-class 是哪个组件的属性?
4.怎么定义 vue-router 的动态路由? 怎么获取传过来的值
5.vue-router 有哪几种导航钩子?
6.$route 和 $router 的区别
7.vue-router响应路由参数的变化
8.vue-router传参
9.vue-router的两种模式
10.vue-router实现路由懒加载（ 动态加载路由 ）

vue常见面试题

1.vue优点
2.vue父组件向子组件传递数据?
3.子组件像父组件传递事件
4.v-show和v-if指令的共同点和不同点
5.如何让CSS只在当前组件中起作用
6.<keep-alive></keep-alive>的作用是什么?
7.如何获取dom
8.说出几种vue当中的指令和它的用法?
9. vue-loader是什么？使用它的用途有哪些?
10.为什么使用key
11.axios及安装
12.axios解决跨域

13.v-modal的使用
14.scss的安装以及使用
15. 请说出vue.cli项目中src目录每个文件夹和文件的用法?
16.分别简述computed和watch的使用场景
17.v-on可以监听多个方法吗
18.$nextTick的使用
19.vue组件中data为什么必须是一个函数
20.vue事件对象的使用
21 组件间的通信
22.渐进式框架的理解
23.Vue中双向数据绑定是如何实现的
24.单页面应用和多页面应用区别及优缺点
25.vue中过滤器有什么作用及详解
26.v-if和v-for的优先级
27.assets和static的区别
28.列举常用的指令
29.vue常用的修饰符
30.数组更新检测
31.Vue.set视图更新
32.自定义指令详解
33.vue的两个核心点
34.vue和jQuery的区别
35 引进组件的步骤
36.Vue-cli打包命令是什么？打包后悔导致路径问题，应该在哪里修改
37.三大框架的对比
38. 跨组件双向数据绑定
39.delete和Vue.delete删除数组的区别
40.SPA首屏加载慢如何解决
41.Vue-router跳转和location.href有什么区别
42. vue slot
43.你们vue项目是打包了一个js文件，一个css文件，还是有多个文件?
44.vue遇到的坑，如何解决的?
45.Vue里面router-link在电脑上有用，在安卓上没反应怎么解决?
46.Vue2中注册在router-link上事件无效解决方法
47.RouterLink在IE和Firefox中不起作用（路由不跳转）的问题
48.axios的特点有哪些
49.请说下封装 vue 组件的过程?
50.vue 各种组件通信方法（父子 子父 兄弟 爷孙 毫无关系的组件）
51.params和query的区别
52. vue mock数据

53 vue封装通用组件
54.vue初始化页面闪动问题
55.vue禁止弹窗后的屏幕滚动
56.vue更新数组时触发视图更新的方法
57.vue常用的UI组件库
58. vue如何引进本地背景图片
59. vue如何引进sass
60.vue修改打包后静态资源路径的修改

vuex常见面试题

1.vuex是什么？怎么使用？哪种功能场景使用它?
2.vuex有哪几种属性
3.不使用Vuex会带来什么问题
4.Vue.js中ajax请求代码应该写在组件的methods中还是vuex的actions中?
5.vuex一个例子方法
6.Vuex中如何异步修改状态
7.Vuex中actions和mutations的区别

vue项目实战

1.顶部悬停效果
2.电话本列表效果（ 右边字母分类 上下滑动 旁边字母显示高亮）
3.vue做代理
4.Vue路由切换时的左滑和右滑效果示例

ES6面试题

ES6新增方法面试题

1.let const var比较
2.反引号（`）标识
3.函数默认参数
4.箭头函数
5.属性简写
6.方法简写
7.Object.keys()方法，获取对象的所有属性名或方法名
8.Object.assign ()原对象的属性和方法都合并到了目标对象

9.for...of 循环
10.import和export
11.Promise对象
12.解构赋值
13.set数据结构（可用于快速去重）
14.Spread Operator 展开运算符(...)
15.字符串新增方法

ES6数组面试题

1.forEach()
2.map()
3.filter()
4.reduce()
5.some()
6.every()
7.all()方法

ES6编程题

1.使用解构，实现两个变量的值的交换
2.利用数组推导，计算出数组 [1,2,3,4] 每一个元素的平方并组成新的数组。
3.使用ES6改下面的模板
4.把以下代码使用两种方法，来依次输出0到9?

react面试题
react生命周期面试题

1.react 生命周期函数
2.react生命周期中，最适合与服务端进行数据交互的是哪个函数
3.运行阶段生命周期调用顺序
4.shouldComponentUpdate 是做什么的，（react 性能优化是哪个周期函数？）
5.指出(组件)生命周期方法的不同

react 基础面试题

1.React 中 keys 的作用是什么?
2.React 中 refs 的作用是什么?
3.React 中有三种构建组件的方式

4.调用 setState 之后发生了什么?
5.react diff 原理（常考，大厂必考）
6.为什么建议传递给 setState 的参数是一个 callback 而不是一个对象
7.除了在构造函数中绑定 this，还有其它方式吗
8.setState第二个参数的作用
9.(在构造函数中)调用 super(props) 的目的是什么
10.简述 flux 思想
11.在 React 当中 Element 和 Component 有何区别?
12.描述事件在 React 中的处理方式。
13.createElement 和 cloneElement 有什么区别?
14.如何告诉 React 它应该编译生产环境版本?
15.Controlled Component 与 Uncontrolled Component 之间的区别是什么?

react组件面试题

1.展示组件(Presentational component)和容器组件(Container component)之间有何不同
2.类组件(Class component)和函数式组件(Functional component)之间有何不同
3.(组件的)状态(state)和属性(props)之间有何不同
4.何为受控组件(controlled component)
5.何为高阶组件(higher order component)
6.应该在 React 组件的何处发起 Ajax 请求
7.react中组件传值
8.什么时候在功能组件(Class Component)上使用类组件(Functional Component)?
9.受控组件(controlled component)与不受控制的组件(uncontrolled component)有什么区别?
10.react 组件的划分业务组件技术组件?

redux面试题

1.redux中间件
2.redux有什么缺点
3.了解 redux 么，说一下 redux 把

react性能比较面试题

1.vue和react的区别
2.react性能优化的方案
3.React 项目用过什么脚手架

4.介绍一下webpack webpack

5.如果你创建了类似于下面的 Twitter 元素，那么它相关的类定义是啥样子的?

6.为什么我们需要使用 React 提供的 Children API 而不是 JavaScript 的 map?

js面试题

1.简述同步和异步的区别

答:

同步： 同步表示会有一个等待的过程，必须要等上一步的操作完成之后才能进行下一步

异步：异步与同步正好相反，不需要有等待的过程，不必等待上一步操作完成，就可以进行下一步

2.怎么添加、移除、复制、创建、替换和查找节点

appendChild() removeChild() cloneNode() replaceChild() createElement()

3.实现一个函数clone 可以对Javascript中的五种主要数据类型（Number、string、Object、Array、Boolean）进行复制

```
function clone (param) {
    let item;
    if (isclass(param) === 'Array') {
        item = param.map((obj) => {
            return obj
        })
    } else if (isclass(param) === 'Object') {
        let obj = {}
        for(key in param) {
            if（param. hasOwnProperty(key)） {
                if(param[key] && isclass(param[key]) === 'Object') {
                    obj[key] = clone(param[key])
                } else {
                    obj[key] = param[key]
                }
            }
        }
        item = obj
    } else {
        item === param
    }
    return item
}
function isclass (o) {
```

```
    if (o === null) return ‘Null’
    if (o === undefined) return ‘Undefined’
    return Object.prototype.toString.call(o).slice(8, -1)
}
```

4.如何消除一个数组里面重复的元素

```
(1) set()
(2) function (pram) {
let  arr = []
for(let i = 0; i<param.length; i++){
    let  item = param[i]
    let isTrue = true
     for(let a = 0;a < arr.length; a++) {
            let obj = arr[a]
            if(item ===obj) {
                isTrue = false
             }
      }
       if(!isTrue) {
            arr.push(item)
        }
    }
}
console.log(arr)
```

5.写一个返回闭包的函数

```
function(){
    let a = 10
    let fun = function () {
          return a
    }
    return fun
}
```

6.使用递归完成1到100的累加

7.Javascript有哪几种数据类型?

null, undefined，Boolean, String, Number，object， array, set, symbol, map

8.如何判断数据类型

Object.prototype.toString.call()

9.console.log(1+’2')和console.log(1-'2')的打印结果

12, -1

10.Js的事件委托是什么，原理是什么
把事件委托给父元素上，利用冒泡的原理，当子元素被点击时，由于冒泡原理，事件会冒泡到父元素上，父元素上的事件就会被触发。

11.如何改变函数内部的this指针的指向
call():
语法：fun.call(this, arg1, arg2, arg3)
第一个参数作用来指定函数内部的this指向，后面的参数是函数执行时所用的参数
apply()：
语法：fun.apply(this, [])
第一个参数用来指定函数内部的this指向
第二个参数要求是一个数组或者是伪数组，apply会把它平铺然后传入调用的函数
bind()：
在Function的prototype上有个新方法bind(),他返回一个新函数：
语法：新函数 = 函数.bind(this)
call, apply, bind的区别
相同点：都可以改变this的指向
不同点：call和apply是一次性改变this指向，而bind是永久性

12.列举几种解决跨域问题的方式，且说明原理
1)Jsonp：通过动态创建script标签请求一个带参网站实现跨领通信
2)CORS：服务端设置Access-Control-Allow-Origin,需要携带cookie,设置withCredentials为true
3)document.domain + iframe：此方法仅限主域相同，子域名不同，俩个页面都通过js强制设置document.domain为基础域名来实现跨域问题。
4)location.href + iframe：a欲与b跨域相互通信，通过中间页c来实现。 三个页面，不同域之间利用iframe的location.hash传值，相同域之间直接js访问来通信。
5)window.name + iframe：name的值在不同页面甚至不同域名加载后值依旧存在，并且可以支持超长值。
6)postMassage
7)webSocket

13.谈谈垃圾回收机制的方式及内存管理
1)标记清除：大部分的浏览器都采用这种方法，当变量进入执行环境（函数中申明，执行时），垃圾回收器标记为“进入环境”，当环境离开环境时（函数执行完成），标记为“离开环境”，在离开环境后，还有的变量时需要被删除的，垃圾收集器给内存中的所有变量都加上标记，然后去掉环境中的变量以及被环境中的变量引用的变量的标记。在此之后再被加上的标记的变量即为需要回收的变量，因为环境中的变量已经无法访问到这些变量。

2)引用计数：这种方式常常会引起内存泄漏，低版本的IE使用这种方式。机制就是跟踪一个值的引用次数，当声明一个变量并将一个引用类型赋值给该变量时该值引用次数加1，当这个变量指向其他一个时该值的引用次数便减一。当该值引用次数为0时就会被回收。该方式会引起内存泄漏的原因是它不能解决循环引用的问题。

14.写一个function ， 清除字符串前后的空格

```
Object.prototype.trim = function (param) {
        if(!param) return
        return param.replace(/(^\s*)|(\s*$)/g, "")
}
```

15.js实现继承的方法有哪些

16.判断一个变量是否是数组，有哪些办法

Object.prototype.toString.call()—>“[Array object]”

17.let ， const ， var 有什么区别

1)var声明的变量存在变量提升，会造成全局污染

2)let 声明的变量为是块级作用于，存在暂时性死区且不可重复申明

3)const声明一个只读常量，且不可重复声明，但声明引用类型是可以修改的，const声明的引用类型变量只保障指针不变

18.箭头函数与普通函数有什么区别

1)this的指向不同，箭头函数this指向上下文

2)箭头函数是匿名函数，不能作为构造函数，不能用new

3)箭头函数绑定的参数是rest,而普通函数绑定的参数是arguments

4)箭头函数在使用apply和call的时候，只允许传入参数，对this没有影响

5)箭头函数没有原型

6)箭头函数不能作为generator函数，不能使用yield关键字

19.随机取1-10之间的整数

Math.ceil(Math.random() * 10)

20.new操作符具体干了什么

1) 创建了一个空对象

2) 将构造函数的作用域给了新对象

3) 执行构造函数中的代码

4) 返回新对象

21.Ajax原理

Ajax的工作原理，相当于是在客户端和服务器端增加了一个中间层，使得用户的操作和服务器响应异步化，并不是所有的用户操作都会提交到服务器，像一些数据验证或

者是数据处理等都是提交Ajax引擎来完成，只有真正的要从服务端获取数据的时候再由Ajax引擎代为向服务器发送请求。

22.模块化开发怎么做

23.异步加载Js的方式有哪些

1)defer支持IE

2)async

3)创建script标签插入到dom中，加载完成后callback

24.xml和 json的区别

26.常见web安全及防护原理

27.用过哪些设计模式

28.为什么要同源限制

同源主要是表示协议，域名，端口的一致，限制同源主要是为了安全考虑。

30.javascript有哪些方法定义对象

1) 工厂模式

```
function fun (name, age) {
      var obj = new Object()
          obj.name = name
          obj.age = age
}
let newObj = fun('xiaoming', 32)
```

2)构造函数模式

```
function Animal (name, age) {
     this.name = name
     this.age = age
}
let newObj = new Animal('xiaoming', 32)
```

3)原型模式

```
function Student () {}
Student.prototype.name = 'xiaoming'
Student.prototype.age = 32
let newObj = new Student()
```

31.说说你对promise的了解

32.谈谈你对AMD、CMD的理解

33.web开发中会话跟踪的方法有哪些

34.介绍js有哪些内置对象?

Number, String, Boolean, Function,Error, Array, Date, Math, RegExp, Global, Object

37.eval是做什么的?

eval() 函数可计算某个字符串，并执行其中的的 JavaScript 代码。
该方法只接受原始字符串作为参数，如果 string 参数不是原始字符串，那么该方法将不作任何改变地返回。因此不要为 eval() 函数传递 String 对象来作为参数。
如果试图覆盖 eval 属性或把 eval() 方法赋予另一个属性，并通过该属性调用它，则 ECMAScript 实现允许抛出一个 EvalError 异常。

38.null，undefined 的区别?
null 表示一个变量将要保存的是一个对象，但是还没有真正保存的对象
undefined 表示一个变量已经声明，但是并没有赋值

39.[“1”, “2”, “3”].map(parseInt) 答案是多少?
[1, NaN, NaN]

40.javascript 代码中的”use strict”;是什么意思？使用它区别是什么?
use strict是es5添加的JS运行模式，主要是作用是为了使的编码更规范，不要出现怪异的方式。

41.js延迟加载的方式有哪些?
1)将js放到最后加载
2)setTimeout
3)defer和async

42.defer和async
1)在script标签里没有写defer或者是async的时候，会立即执行脚本，所以要把script标签放在body后面
2)script标签里设置defer，加载后续文档元素的过程和script.js加载是并行的，但是script.js的执行要在所有元素解析完成之后，DOMContentLoaded事件触发之前完成，并且多个defer会安照顺序进行加载。
3)有 async，加载和渲染后续文档元素的过程将和 script.js 的加载与执行并行进行（异步）。

45.ECMAScript6 怎么写class么?

```
class App {
    constructor(name, age) {
        this.name = name
        this.age = age
    }
    say() {
        console.log(111)
    }
```

}
let obj = new App('xiaoming', 32)
47.函数防抖，节流的原理
函数节流是指一定时间内js方法只跑一次。比如人的眨眼睛，就是一定时间内眨一次。这是函数节流最形象的解释。函数节流应用的实际场景，多数在监听页面元素滚动事件的时候会用到。因为滚动事件，是一个高频触发的事件。
函数防抖是指频繁触发的情况下，只有足够的空闲时间，才执行代码一次。比如生活中的坐公交，就是一定时间内，如果有人陆续刷卡上车，司机就不会开车。只有别人没刷卡了，司机才开车。函数防抖的应用场景，最常见的就是用户注册时候的手机号码验证和邮箱验证了。只有等用户输入完毕后，前端才需要检查格式是否正确，如果不正确，再弹出提示语。函数防抖的实现重点，就是巧用`setTimeout`做缓存池，而且可以轻易地清除待执行的代码。其实，用队列的方式也可以做到这种效果。这里就不深入了。

48.原始类型有哪几种？null是对象吗?
Boolean，String，Number, Undefined, Null ,
Null严格来说不是对象，申明变量的值为null表示这个变量要保存一个对象，而null保存了指向要保存的对象的地址

49.为什么console.log(0.2+0.1==0.3) //false

53.== 和 ===的区别
==：只要求值相等
===：要求值相同，且数据类型相同
54.什么是闭包

```
function fun () {
    var a = 1
    return function () {
        return a
    }
}
```

闭包的用途：
1）读取函数内部的变量
2）闭包中的变量会一直保留在内存中

55.JavaScript原型，原型链？有什么特点?
56.typeof()和instanceof()的用法区别
typeof（） 是一个一元运算，放在一个运算数之前，运算数可以是任意类型。
它返回值是一个字符串，该字符串说明运算数的类型。，typeof一般只能返回如下几个结果：number,boolean,string,function,object,undefined。 我们可以使用typeof来获取一个变量是否存在，如if(typeof a!="undefined"){alert("ok")}，而不要去使用if(a)因为如果a

不存在（未声明）则会出错，对于Array,Null，DOM对象等特 殊对象使用typeof一律返回object，这正是typeof的局限性。

instanceof 用于判断一个变量是否某个对象的实例，如var a=new Array();alert(a instanceof Array);会返回true，同时alert(a instanceof Object)也会返回true;这是因为Array是object的子类。再如：function test(){};var a=new test();alert(a instanceof test)会返回

57$(document).ready()方法和window.onload有什么区别?
59.为什么会出现setTimeout倒计时误差？如何减少
60.谈谈你对JS执行上下文栈和作用域链的理解
62.prototype 和 proto 区别是什么?
64.取数组的最大值（ES5、ES6）
排序

65.ES6新的特性有哪些?
Default Parameters（默认参数） in ES6
Template Literals （模板文本）in ES6
Multi-line Strings （多行字符串）in ES6
Destructuring Assignment （解构赋值）in ES6
Enhanced Object Literals （增强的对象文本）in ES6
Arrow Functions （箭头函数）in ES6
Promises in ES6
Block-Scoped Constructs Let and Const（块作用域构造Let and Const）
Classes（类） in ES6
Modules（模块） in ES6

66.promise 有几种状态, Promise 有什么优缺点 ?
Promise的状态：padding fulfilled reject
缺点：promise会有延时问题
promise在padding状态的时候，无法得知进展在哪一步
promise每次调用then方法的时候都会返回一个promise，会出现内存浪费
promise会吞噬内部抛出的错误，如果最后一个then方法内有错误不会抛出

67.Promise构造函数是同步还是异步执行，then呢 ?promise如何实现then处理 ?
68.Promise和setTimeout，Async/Await的区别 ?
1)首先了解js的宏观任务和微观任务
宏观任务：由浏览器宿主发起执行的为宏观任务，如：setTimeout, setInterval等
微观任务：由js引擎发起的任务，如：promise，Object.observe等
微观任务要优先于宏观任务执行。所以，promise要优先于setTimeout执行。
2)promise本身是同步立即执行的函数，当在executor中执行resolve或者reject的时候，此时是异步操作，会先执行then/catch等，当主栈完成之后，才去调用resolve/reject中存

放的方法执行。当js执行到主线程执行到promise对象时，promise1.then()的回调是一个task，
promise1是resolve或者rejected：那么这个task就会被放入到当前事件循环回合的microtask queue,promise1是pending：这个task就是会放入到事件循环到未来的某个（可能下一个）回合的microtask queue中
setTimeout的回调也是个task，他会被放入到macrotask queue
async/await: async函数返回一个promise对象，当函数执行到时候，一旦遇到await就会先返回，等到触发的异步操作完成，再执行函数体内后面的语句，可以理解为是让线程跳出了async的函数体。await的含义为等待，也就是async函数需要等待await后的函数执行完成后并且有了结果返回结果（promise对象）之后，才能继续执行下面的代码，await通过返回一个promise来实现同步的效果。
69.如何实现 Promise.all？

70.如何实现 Promise.finally？

71.如何判断img加载完成
1)轮询不断的监听img的 complete 属性，如果complete的值为true，则表明图片已经加载完成，停止轮询。
2)img的load事件中证明图片加载完成
3)readyState为complete和loaded则表明图片已经加载完毕，在IE6-IE10支持该事件，其他不支持。

72.如何阻止冒泡?
e.preventDefault()

73.如何阻止默认事件?
e.stopPropagation()

74.ajax请求时，如何解释json数据
JSON.parse()　JSON.stringfy()

75.json和jsonp的区别?
json是一种数据格式
Jsonp是一种传输数据的方式

76.如何用原生js给一个按钮绑定两个onclick事件?
我是按钮
var btn=document.getElementByClassName('btn')
btn. addEventListener('click', fun1)
btn. addEventListener('click', fun2)
function fun1 () {}
function fun2 () {}

77.拖拽会用到哪些事件
onmousedown onmousemove onmouseup

78.document.write和innerHTML的区别
document.write会直接写入内容流，会导致页面重绘
innerHTML是写入某个DOM节点，不会导致页面重绘

79.jQuery的事件委托方法bind 、live、delegate、on之间有什么区别?
80.浏览器是如何渲染页面的?
1)用户输入网址（假设是个 HTML 页面，第一次访问，无缓存情况），浏览器向服务器发出HTTP请求，服务器返回 HTML 文件；（善用缓存，减少HTTP请求，减轻服务器压力）

2)浏览器载入 HTML 代码，发现 <head> 内有一个 <link> 引用外部 CSS 文件,则浏览器立即发送CSS文件请求，获取浏览器返回的CSS文件;（CSS文件合并，减少HTTP请求）

3)浏览器继续载入 HTML 中 <body> 部分的代码，并且 CSS 文件已经拿到手了，可以开始渲染页面了;（CSS文件需要放置最上面，避免网页重新渲染）

4)浏览器在代码中发现一个 <img> 标签引用了一张图片，向服务器发出请求。此时浏览器不会等到图片下载完，而是继续渲染后面的代码；（图片文件合并，减少HTTP请求）

5) 服务器返回图片文件，由于图片占用了一定面积，影响了后面段落的排布，因此浏览器需要回过头来重新渲染这部分代码；（最好图片都设置尺寸，避免重新渲染）

6) 浏览器发现了一个包含一行 JavaScript 代码的 <script> 标签，会立即运行该js代码；（script最好放置页面最下面）

7) js脚本执行了语句，它令浏览器隐藏掉代码中的某个 ,突然就少了一个元素，浏览器不得不重新渲染这部分代码；（页面初始化样式不要使用js控制）

8) 终于等到了 </html> 的到来，浏览器泪流满面……

9) 等等，还没完，用户点了一下界面中的“换肤”按钮，JavaScript 让浏览器换了一下 <link> 标签的 CSS 路径;

10) 浏览器召集了在座的各位 们，“大伙儿收拾收拾行李，咱得重新来过……”，浏览器向服务器请求了新的CSS文件，重新渲染页面。

81.$(document).ready()方法和window.onload有什么区别?

1).执行时机
window.onload方法是在网页中的所有的元素（包括元素的所有关联文件）都完全加载到浏览器之后才执行。这种方式有一个很大的优点:不用考虑DOM元素加载的顺序。而通过jQuery中的$(document).ready()方法注册的事件处理程序，只要在DOM完全就绪时，就可以调用了，比如一张图片只要标签完成，不用等这个图片加载完成，就可以设置图片的宽高的属性或样式等。这种方式优于onload()事件在于:$(document).ready()可以在页面没有完全下载时,操作页面的DOM元素.

2)使用次数
window.onload不能同时编写多个，如果有多个window.onload方法，只会执行一个
$(document).ready()可以同时编写多个，并且都可以得到执行

3)简化写法
window.onload没有简化写法
$(document).ready(function(){})可以简写成$(function(){});

82. get请求和post请求有区别吗?
1)get请求的参数是在url中可以看到，而post请求的参数是在body中，不被看到
2)get请求携带的数据量比较小，由于url长度的限制只能携带1024字节，而post的比较大2mb左右
3)get请求会有缓存的问题，而post不用担心缓存问题
4)post请求必须要设置content-type
5)因为get请求的参数会被浏览器缓存起来，所以get请求的安全性比较低

83.对前端路由的理解？前后端路由的区别?
前端路由，说白了就是前端页面状态的管理器，可以不通过向后端发送请求，而使用前端技术实现前端多个页面的效果。路由主要有俩种:
1)hash模式
页面中通过锚点定位的原理实现页面无刷新跳转，触发URL中'#'后面的部分，同时触发在全局window上的hashChange事件，这样在页面锚点hash改变为某个预备值的时候，通过代码触发对应的DOM改变，就可以实现页面的基本路由了。
2)history模式
history模式主要是通过为浏览器的history对象添加扩展方法，一般是用来解决Ajax请求无法通过回退按钮回到请求之前的状态。浏览器访问一个页面时，当前地址的状态信息会被压入历史栈,当调用history.pushState()方法向历史栈中压入一个新的state后，历史栈顶部的指针是指向新的state的。可以将其作用简单理解为 假装已经修改了url地址并进行了跳转 ,除非用户点击了浏览器的前进,回退,或是显式调用HTML4中的操作历史栈的方法，否则不会触发全局的popstate事件。

对比	hash路由	History API 路由

url字符串	丑	正常
命名限制	通常只能在同一个`document`下进行改变	url地址可以自己来定义，只要是同一个域名下都可以，自由度更大
url地址变更	会改变	可以改变，也可以不改变
状态保存	无内置方法，需要另行保存页面的状态信息	将页面信息压入历史栈时可以附带自定义的信息
参数传递能力	受到url总长度的限制，	将页面信息压入历史栈时可以附带自定义的信息
实用性	可直接使用	通常服务端需要修改代码以配合实现
兼容性	IE8以上	IE10以上

84.手写一个类的继承

```
function Animal (name) {
        this.name = name
        this.age = 5
}
var dog =. new Animal(‘xiaoming’)
console.log(dog)
```

85.XMLHttpRequest：XMLHttpRequest.readyState;状态码的意思

200: 请求成功

500: 服务器故障

404: 资源不存在

.

.

.

5.什么叫优雅降级和渐进增强
6.px和em的区别
7.HTML5 为什么只写
8.Http的状态码有哪些
9.一次完整的HTTP事务是怎么一个过程
10.HTTPS是如何实现加密
11.浏览器是如何渲染页面的
12.浏览器的内核有哪些？分别有什么代表的浏览器
13.页面导入时，使用link和@import有什么区别
14.如何优化图像，图像格式的区别
15.列举你了解Html5. Css3 新特性
16.可以通过哪些方法优化css3 animation渲染
17.列举几个前端性能方面的优化
18.如何实现同一个浏览器多个标签页之间的通信
19.浏览器的存储技术有哪些
20.css定位方式
21.尽可能多的写出浏览器兼容性问题
22.垂直上下居中的方法
23.响应式布局原理
25.清除浮动的方法
26.http协议和tcp协议
27.刷新页面，js请求一般会有哪些地方有缓存处理
28.如何对网站的文件和资源进行优化
29.你对网页标准和W3C重要性的理解
30.Http和https的区别
31.data-属性的作用
32.如何让Chrome浏览器显示小于12px的文字
33.哪些操作会引起页面回流（Reflow）
34.CSS预处理器的比较less sass
35.如何实现页面每次打开时清除本页缓存
36.什么是Virtual DOM,为何要用Virtual DOM
37.伪元素和伪类的区别
38.http的几种请求方法和区别
39.前端需要注意哪些SEO
40.的title和alt有什么区别
41.从浏览器地址栏输入url到显示页面的步骤
42.如何进行网站性能优化
43.语义化的理解
44.HTML5的离线储存怎么使用，工作原理能不能解释一下？

45.浏览器是怎么对HTML5的离线储存资源进行管理和加载的呢
46.iframe有那些缺点?
47.WEB标准以及W3C标准是什么?
48.Doctype作用? 严格模式与混杂模式如何区分？它们有何意义?
49.HTML全局属性(global attribute)有哪些
50.Canvas和SVG有什么区别?
51.如何在页面上实现一个圆形的可点击区域?
52.网页验证码是干嘛的，是为了解决什么安全问题
53.请描述一下 cookies，sessionStorage 和 localStorage 的区别?
54. CSS选择器有哪些？哪些属性可以继承?
55.CSS优先级算法如何计算?
56.CSS3有哪些新特性?
57.请解释一下CSS3的flexbox（弹性盒布局模型）,以及适用场景?
58.用纯CSS创建一个三角形的原理是什么?
59.常见的兼容性问题?
60.为什么要初始化CSS样式
61.absolute的containing block计算方式跟正常流有什么不同?
62.CSS里的visibility属性有个collapse属性值？在不同浏览器下以后什么区别?
63.display:none与visibility：hidden的区别?
64.position跟display、overflow、float这些特性相互叠加后会怎么样?
65.对BFC规范(块级格式化上下文：block formatting context)的理解?
66.为什么会出现浮动和什么时候需要清除浮动？清除浮动的方式?
67.上下margin重合的问题
68. 设置元素浮动后，该元素的display值是多少?
69.移动端的布局用过媒体查询吗?
70.CSS优化、提高性能的方法有哪些?
71.浏览器是怎样解析CSS选择器的?
72.在网页中的应该使用奇数还是偶数的字体？为什么呢?
73.margin和padding分别适合什么场景使用?
74.元素竖向的百分比设定是相对于容器的高度吗?
75.全屏滚动的原理是什么？用到了CSS的哪些属性?
76.什么是响应式设计？响应式设计的基本原理是什么？如何兼容低版本的IE?
77. 视差滚动效果?
78.::before 和 :after中双冒号和单冒号有什么区别？解释一下这2个伪元素的作用
79.让页面里的字体变清晰，变细用CSS怎么做?
80. position:fixed;在android下无效怎么处理?
81.如果需要手动写动画，你认为最小时间间隔是多久，为什么?
82.li与li之间有看不见的空白间隔是什么原因引起的？有什么解决办法?
83.display:inline-block 什么时候会显示间隙?

84. 有一个高度自适应的div，里面有两个div，一个高度100px，希望另一个填满剩下的高度
85.png、jpg、gif 这些图片格式解释一下，分别什么时候用。有没有了解过webp?
86.style标签写在body后与body前有什么区别?
87.CSS属性overflow属性定义溢出元素内容区的内容会如何处理?
88.阐述一下CSS Sprites
89. 一行或多行文本超出隐藏

微信小程序开发（持续更新）

初识小程序

1.注册小程序
2.微信开发者工具
3.小程序与普通网页开发的区别
4.小程序尺寸单位rpx
5.样式导入（WeUI for）
6.选择器
7.小程序image高度自适应及裁剪问题
8.微信小程序长按识别二维码
9.给页面加背景色
10.微信小程序获取用户信息
11.代码审核和发布
12.小程序微信认证
13.小程序申请微信支付
14.小程序的目录解构及四种文件类型
15.小程序文件的作用域
16.小程序常用组件
    1.view
    2.scroll-view
    3.swiper组件
    4.movable-view
    5.cover-view
    6.cover-image

小程序基础

17.授权得到用户信息

18.数据绑定
19.列表渲染
20.条件渲染
21.公共模板建立
22.事件及事件绑定
23.引用
24.页面跳转
  1.wx.switchTab
  2.wx.reLaunch
  3.wx.redirectTo
  4.wx.navigateTo
  5.wx.navigateBack
25.设置tabBar
26.页面生命周期
27.转发分享

小程序高级

28.request请求后台接口
29.http-promise 封装
30.webview
31.获取用户收货地址
32.获取地里位置
33.自定义组件
34.微信小程序支付问题

小程序项目实战

35.微信小程序本地数据缓存
36.下拉刷新和下拉加载
37.列表页向详情页跳转（动态修改title）
38.客服电话
39.星级评分组件
40.小程序插槽的使用slot
41.模糊查询
42.wxs过滤
43.小程序动画
44.列表根据索引值渲染
45.小程序动态修改class

46.小程序常用框架
47.参数传值的方法
48.提高小程序的应用速度
49.微信小程序的优劣势
50.小程序的双向绑定和vue的区别
51.微信小程序给按钮添加动画
52.微信小程序的tab按钮的转换
53.微信小程序引进echarts
54.APP打开小程序流程
55.小程序解析富文本编辑器

小程序常见bug

1.域名必须是HTTPS
2. input组件placeholder字体颜色
3. wx.navigateTo无法跳转到带tabbar的页面
4. tabbar在切换时页面数据无法刷新
5.如何去掉自定义button灰色的圆角边框
6.input textarea是APP的原生组件，z-index层级最高
7.一段文字如何换行
8.设置最外层标签的margin-bottom在IOS下不生效
9.小程序中canvas的图片不支持base64格式
10.回到页面顶部
11.wx.setStorageSync和wx.getStorageSync报错问题
12.如何获取微信群名称?
13.new Date跨平台兼容性问题
14.wx.getSystemInfoSync获取windowHeight不准确
15.图片本地资源名称，尽量使用小写命名

移动端热点问题
1. 1px border问题
2.2X图 3X图适配
3.图片在安卓上，有些设备模糊问题
4.固定定位布局 键盘挡住输入框内容
5.click的300ms延迟问题和点击穿透问题
6.phone及ipad下输入框默认内阴影
7.防止手机中页面放大和缩小
8.flex布局

9.px、em、rem、%、vw、vh、vm这些单位的区别
10. 移动端适配- dpr浅析
11.移动端扩展点击区域
12 上下拉动滚动条时卡顿、慢
13 长时间按住页面出现闪退
14. ios和android下触摸元素时出现半透明灰色遮罩
15. active兼容处理 即 伪类：active失效
16.webkit mask兼容处理
17. pc端与移动端字体大小的问题
18. transiton闪屏
19.圆角bug
20.如何解决禁用表单后移动端样式不统一问题?

js常用插件

轮播图插件
二级城市插件
三级城市插件
文字滑动效果
手风琴效果
视频播放插件
弹层插件
百度编辑器
ACE编辑器（轻巧）
上传图片（裁剪）
页面加载效果
全选反选各种效果
京东楼层效果
懒加载

快速建站（全栈）
dedecms（文章累）
discuz（论坛）
ecshop（电商）
PHPEMS（考试）