面试汇总

致保科技-一面（通过）【210901】

1. ES6 中你用到了那些新特性

let、const

class extends

promise + async/await

解构赋值

map, filter, reduce

ES Module

模板字符串

2. 用过哪些性能优化手段

https://docs.qq.com/doc/DQVFvcmduSlZLQmRz

3. 项目中哪些方案是你设计的，可以具体说说么

- 加密模块：AES 对称加密、RSA 非对称加密
- 埋点模块：hybrid SDK、JS SDK
- dsbridge 模块：原生 SDK 联调、封装
- arms 监控：环境变量配置
- H5 收银台：
  - 微信支付（场景：APP-APP 支付、H5-H5 支付、公众号-JSAPI 支付、扫码-native 支付）
  - 支付宝支付（场景：APP-APP 支付、H5-手机网站支付、扫码-当面付）

- 推荐模块：submodule -> cdn 引入
- 缓存策略配置：静态资源强缓存
- eslint 与 stylelint 规范配置
- 预渲染、骨架屏方案

4. vue 自定义指令用过么，有哪些应用场景

https://www.cnblogs.com/lzq035/p/14183553.html

v-debounce

v-LazyLoad

v-draggable

v-focus

5. 深拷贝与浅拷贝了解过么，怎么实现深拷贝

浅拷贝：Object.assign、Array.concat

深拷贝：JSON.parse(JSON.stringify(obj))

问题：1.会忽略 undefined 和 symbol 2.不能序列化函数 3.不能解决对象的循环引用

MessageChannel 消息通道 实现深拷贝（不包含函数）

lodash 深拷贝函数

6. 看你项目用了 CDN 加速，CDN 回源策略了解过么

DNS 原理：https://blog.csdn.net/xiangzhihong8/article/details/83147542

常规的 CDN 都是回源的。即：当有用户访问某一个 URL 的时候，如果被解析到的那个 CDN 节点没有缓存响应的内容，或者是缓存已经到期，就会回源站去获取。如果没有人访问，那么 CDN 节点不会主动去源站拿的。

7. 小程序授权流程，拿到用户的什么信息

openId

8. webpack 常用的 loader 和 plugin，loader 和 plugin 区别

参考：https://www.jianshu.com/p/6397d692f61f

loader：

专注于转化文件（transform）这一个领域，完成压缩，打包，语言翻译。【仅仅只是为了打包】。运行在打包文件之前（loader 为在模块加载时的预处理文件）

【css-loader】和【style-loader】【sass-loader】模块是为了打包 css 的

【babel-loader】和【babel-core】模块时为了把 ES6 的代码转成 ES5

【url-loader】和【file-loader】是把图片进行打包的。

【ts-loader】

【eslint-loader】

plugin：

功能更丰富，在整个编译周期都起作用

【commons-chunk-plugin】：提取公共代码

【UglifyJS】：代码混淆压缩-webpack4 内置 默认开启

【DllPlugin】：代码切割-内置

【webpack-bundle-analyzer】：打包分析

【optimize-css-assets-webpack-plugin】：css 去重

【HappyPack 】多线程打包

9. 项目安全做了哪些处理

xss 跨站脚本漏洞：字符串转义、csp 白名单（Content-Security-Policy）

CSRF 跨站请求伪造：验证 Referer、Token 认证、

点击劫持：防止 iframe 嵌套 X-FRAME-OPTIONS

中间人攻击：使用 https

10.for..in 和 for..of 有什么区别,foreach

https://zhuanlan.zhihu.com/p/161892289

得物 APP-一面（通过）【210901】

1. v-model 双向绑定原理【vue】

2. vue 中的 data 为什么是个 function【vue】

3. vue-router 的几种模式，原理是什么

4. 观察者模式，发布订阅模式，两者有何区别【vue】

https://zhuanlan.zhihu.com/p/51357583

5. nextTick 原理【vue】

6. vue 中 key 的作用【vue】

7. diff 算法具体怎么对比的【vue】

8. watch 与 computed 应用场景，区别，怎么实现的数组 watch【vue】

区别：https://www.cnblogs.com/jiajialove/p/11327945.html

深度监听数组与对象：https://www.cnblogs.com/neo-java/p/11377658.html

9. es6 用了啥，详细说说 promise 原理【es6】

10.object.defineProtoype 与 proxy 有啥不同【es6】

https://www.jianshu.com/p/7ced8c744477

11.箭头函数与普通函数区别【es6】

12.eventLoop

13.webpack 常用 loader 与 plugin

14.缓存策略

15.常见的 http 状态码，206 知道么

16.事件委托

17.BFC（块级格式化上下文）

https://blog.csdn.net/sinat_36422236/article/details/88763187

overflow、flex、float、position

1. 利用 BFC 避免 margin 重叠。2. 自适应两栏布局 3. 清除浮动。

18.flex 常用属性，flex:1 (1 1 0%) flex:auto(1 1 auto) flex:none(0 0 auto)

第一个参数表示: flex-grow 定义项目的放大比例，默认为 0，即如果存在剩余空间，也不放大

第二个参数表示: flex-shrink 定义了项目的缩小比例，默认为 1，即如果空间不足，该项目将

缩小

第三个参数表示: flex-basis 给上面两个属性分配多余空间之前，计算项目是否有多余空间，默认值为 auto，即项目本身的大小

19.深拷贝与浅拷贝

得物 APP-二面（通过）【210905】

1. 埋点，请求 1*1 图片有什么好处，跨域问题，跨域怎么携带 cookie

https://blog.csdn.net/a276598490/article/details/101481489

跨域解决方案 https://panjiachen.github.io/vue-element-admin-site/zh/guide/advanced/cors.html

2. vue2 与 vue3 区别有哪些

3. ts 有哪些特性

4. 性能优化，实现秒开什么指标

好买财富-一+二面（offer）【210903】

关于 Vue

1. diff 算法具体对比过程，怎么优化

https://blog.csdn.net/frontend_frank/article/details/114297890

优化-其实是 vue3 的 diff 优化

2. history 与 hash 模式区别

关于 webpack

1. loader 原理，常用 loader，css loader 与 style loader 区别于顺序

2. webpack 5 做了哪些优化

关于项目

1. CDN 原理，图片替换之后缓存问题（名称不变）

2. html 生成 PDF

3. Etag 具体生成方式

4. 阿里乾坤原理

5. taro 原理

6. 服务端渲染

7. 白屏方案

预渲染

服务端渲染

骨架屏

异常捕获

8. 加密模块并发控制（同时生成多个私钥问题）

存在 REA 公钥&&AES 私钥时 return

9. H5 收银台方案设计（链路闭环、获取支付结果）

长轮询、长链接 websocket

远川研究所（offer）-安卓面试【210903】

1. 设计模式

2. 从输入 URL 开始浏览器发生了什么

字节-一面（通过）【210903】

1. Object.defineprototype 能遍历监测数组么？【可以，跟 object 一样】

```
> var a = [1,2,3,4,5]
<- undefined
> a.forEach((item,index)=>{
      Object.defineProperty(a,index,{
  set(newVal){console.log('set====');value = newVal},
  get(){console.log('get----'); return value;},
  }
  )
  })
<- undefined
> a[1]=22
  set====                                                    VM407:3
<- 22
> a[1]
  get----                                                    VM407:4
<- 22
> a.push(11)
<- 6
> a[5]
<- 11
> a[4]
  get----                                                    VM407:4
<- 22
>
```

2. eventBus 原理是什么

发布订阅模式

3. jsbridge 原理

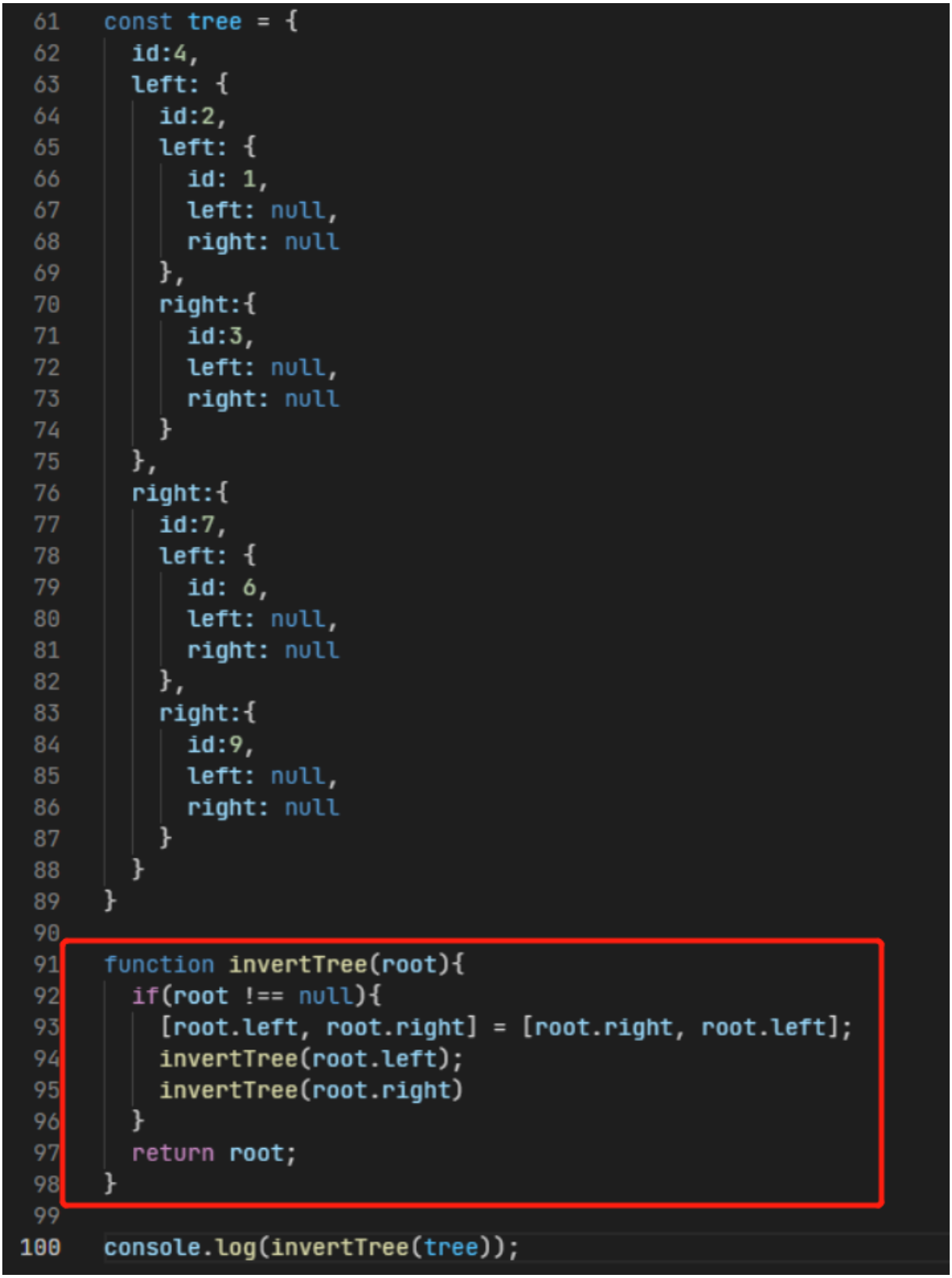
https://www.cnblogs.com/wfblog/articles/12542109.html

4. 有一个很大的 object 需要父组件传递给子组件，怎样可以不对其进行监听

手写题

1. 反转二叉树（所有节点左右节点位置兑换）

https://blog.csdn.net/xiasohuai/article/details/85719229



```
const tree = {
  id:4,
  left: {
    id:2,
    left: {
      id: 1,
      left: null,
      right: null
    },
    right:{
      id:3,
      left: null,
      right: null
    }
  },
  right:{
    id:7,
    left: {
      id: 6,
      left: null,
      right: null
    },
    right:{
      id:9,
      left: null,
      right: null
    }
  }
}

function invertTree(root){
  if(root !== null){
    [root.left, root.right] = [root.right, root.left];
    invertTree(root.left);
    invertTree(root.right)
  }
  return root;
}

console.log(invertTree(tree));
```

2. 实现一个函数，输出一个 div 所有子节点个数-递归所有子节点

function countChildren(dom,sum=0){

```
    if(dom.children && !dom.children.length) return sum;


    // sum+=dom.children.length;
    console.log(dom.children,dom.children.length);
    for(let child of dom.children){
      sum++;//  当前层级节点数
      sum+=countChildren(child)// 子节点节点数
    }
    return sum
  }
```

3. 手写一个发布订阅模式-完善以下代码

```
class event {
      on(eventName,fn){
      //绑定方法
      }
      off(eventName,fn){
      //解除绑定方法
      }
      emit(eventName,data){
      //触发方法
      }
}
```

```
// 简易发布-订阅模式
class Event {
  constructor(){
    this.center = {}
  }

  on(eventName,fn){
    if(this.center[eventName]){
      this.center[eventName].push(fn)
    }else{
      this.center[eventName] =[fn];
    }
  }

  off(eventName,fn){
    this.center[eventName] = this.center[eventName].filter(onFn=>fn != onFn)
  }

  emit(eventName,data){
    this.center[eventName].forEach(fn => {
      fn.call(this,data)
    });
  }
}

const myEvent = new Event();

const eat = (data)=>{ console.log(`吃${data}`) };

myEvent.on('eat',eat)

myEvent.on('eat',(data)=>{
  console.log(`今天吃${data}`);
})

myEvent.emit('eat','饭')

myEvent.off('eat',eat)

myEvent.emit('eat','蘑菇')
```

4. 闭包

```
for(var i=0;i<5;i++){
    setTimeout(()=>{
        console.log(i)
    },i)
}
```

打印？ 5，5，5，5，5

修改之后打印 0，1，2，3，4

```
42  //  let
43  for(let i=0;i<5;i++){
44    setTimeout(()=>{
45      console.log(i)
46    },i)
47  }
48  //  setTimeout传参支持
49  for(var i=0;i<5;i++){
50    setTimeout((i)=>{
51      console.log(i)
52    },i,i)
53  }
54  //  自执行函数
55  for(var i=0;i<5;i++){
56    (function(i){setTimeout((i)=>{
57      console.log(i)
58    },i,i)})(i);
59  }
```

字节-二面（？）【210909】

1. 加密模块，安全相关做过哪些配置

2. iframe 通信，js 如何判断页面被 iframe 嵌套了

https://blog.csdn.net/willspace/article/details/49003963

postmessage/`contentWindow parent.window/top.window`

3. 性能监控与异常监控

4. 盒模型-box-sizing

https://blog.csdn.net/zwkkkk1/article/details/79678177

5. dsbridge 原理，其他 native 通信方式

6. vue-model 原理

手写题

1. 深拷贝-symbol 使用场景，怎样复制 symbol

2. 输入一个数组，查找里面第二大的数字，要求时间复杂度 o(n)

3. 123456789=》一亿两千三百四十五万六千七百八十九

珍岛集团-一面（？）【210904】

1. 深度优先遍历与广度优先遍历

https://blog.csdn.net/weixin_40629549/article/details/104616608

2. nextTick 原理与使用场景

3. 父子组件生命周期执行顺序

https://www.cnblogs.com/caoshouling/p/13403019.html

4. 动态组件 &异步组件

https://cn.vuejs.org/v2/guide/components-dynamic-async.html

5. 自定义指令 钩子函数

6. promise.all

https://zhuanlan.zhihu.com/p/41502945

7. 取消一个 axios

https://segmentfault.com/a/1190000021290514

8. 跨域与代理，代理实现原理

9. display:none; visibility:hidden; opacity:0;

海鼎-牛客网笔试（GG）【210904】

问答

1. eventLoop 输出结果

2. 类型转换

console.log('5'<'10') ('5'<10) +true null== undefined 'bat'>'bbmb'

3. 项目难点亮点

4. 图片预加载

5. watcher 怎么去重-异步更新队列

https://cn.vuejs.org/v2/guide/reactivity.html#%E5%A6%82%E4%BD%95%E8%BF%BD%E8%B8%AA%E5%8F%98%E5%8C%96

6. nextTick 原理

7. this 指向与箭头函数

```
const person1 = {
  name: 'ming',
  say: function () {
    console.log(this.name)
  },
  eat:()=>{
    console.log(this.name)
  }
}
const person2 = {
  name:'hong'
}

person1.say()
person1.say.call(person2)
person1.eat.call(person2)
```

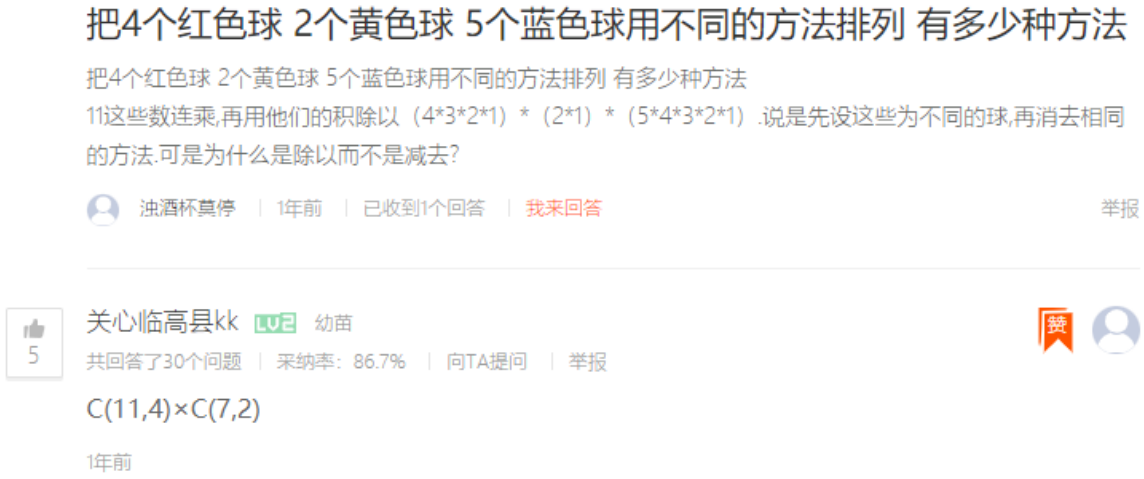
编程

1. 实现一个函数，输入一个数字，将这个数字展开为仅有 1 和 3 存在的数组，有多少这种数组存在？

示例：

输入 4，[1,1,1,1] [1,3] [3,1]，输出 3

输入 6, [1,1,1,1,1,1] [1,1,1,3] [1,1,3,1] [1,3,1,1] [3,1,1,1] [3,3]，输出 6



解答：

（1） 输入 number = x + 3y // 6=0+3*2,x=0,y=2

（2） for(let i=0;i<=y;i++){

number = i*3 + (y-i)*3+x;

i 个 3， (y-i)*3+x 个 1

此时排列组合为 C((y-i)*3+i,i)

// i=0, 0*3+6*1, C(6,6) = 1

// i=1, 1*3+3*1, C(4,1)=4

// i=2, 2*3+0*1, C(2,2)=1

}

2. 实现一个函数，输入一个数组，找出其中可以构成三角形，且边长最长的一组数字

示例：

输入[2,3,4,5,6,10]，输出[3,4,5]

兴业数科

输入一个字符串，判断同一字符间隔最长的字串长度，没有重复字符串返回-1

如：输入 cc，输出 0；输入 abc,输出 0；输入 a12345a，输出 5

满帮

1. nextTick 使用场景，为什么可以在 dom 更新后执行

https://blog.csdn.net/weixin_39889337/article/details/111664456

2. CDN 回源，怎么判断文件已失效

https://www.cnblogs.com/blogbyhuer/p/9335257.html

3. OSS 防盗链与失效时间

4. 原生交互，具体场景调用原生拍照，参数如何回传，base64 跟图片相比大小变化如何，返回 base64 如何实现图片上传

https://www.zhihu.com/question/60593079/answer/247418738

https://blog.csdn.net/jw19950424/article/details/79752713

5. 性能优化与性能指标

6. 异常监控

http://rapheal.sinaapp.com/2014/11/06/javascript-error-monitor/

7. bind 实现原理

[].shift.call(arguments) 和 [].slice.call(arguments) 解释

https://blog.csdn.net/u013946061/article/details/108269650