# 个人简历

## 个人信息


- 姓 名: 郭维开　　岗 位：前端开发工程师
- 性 别：男　　年 龄：33
- 手 机：15605440321　　邮 箱：15605440321@163.com


## 求职意向

前端开发工程师

## 专业技能

- 熟练掌握 HTML、CSS 和 JavaScript 等前端开发基础知识
- 熟练使用 Vue.js和 React 框架及全家桶，并有丰富的项目经验；
- 熟练使用 WebRTC 标准和相关技术，具有丰富的音视频通信开发经验。
- 熟练使用 Webpack 能够基于 Webpack 自定义搭建前端项目, 使用 webpack socket 搭建 web sdk 上传 npm
- 熟练使用 TypeScript , ES6+, Webpack 进行 SDK 开发
- 熟练使用 Socket.io 进行音视频信令传输和 IM 实时通信；
- 熟练使用 Electron 开发 PC 端
- 熟练使用(uni-app)开发小程序
- 熟练使用 nginx 、shell、docker、 jenkins、vpn 等开发工具
- 熟悉 node.js 了解 express 路由,中间件等
- 做过 Flutter 开发

## 工作经历

- 北京乐我无限有限公司(猎豹移动) 2020年 音视频前端开发工程师
- 北京蓝将技术有限公司 2019 年 前端开发工程师
- 北京云中融信有限公司 2017 前端开发工程师
- 北京一雄科技有限公司 2015年 前端开发工程师

## 项目经历

### 1. linkv-octopus-webrtc(北京乐我无限猎豹移动)

- 描述: linkv-octopus-webrtc 是一款实时的音视频互动服务产品，能够为开发者提供便捷接入、高清流畅、多平台互通、低延迟、高并发的音视频服务，可以实现一对多，多对多的实时音视频互 动，秀场直播，视频会议等场.目前使用在乐我 web 端, pc 端, 小墨鱼等产品.
- 技术实现:
  - 技术栈 Webpack, Typescript,EventEmitter, socket.io, Webrtc 等
  - 使用 webpack 作为构建工具进行模块化,库类开发, 构建生成 sdk
  - 各模块之间通过 EventEmitter 进行订阅分发事件通信
  - 引入了 ts 做到静态化检查
  - 基于 socket.io 做信令传输

    - 基于浏览器 webrtc 技术栈实现音频通话和视频通话
    - 视频质量通过埋点上报神策日志和大数据,通过神策和水晶球平台进行数据分析
- 地址:
    - npm 下载 npm i linkv-octopus-webrtc
    - cdn 地址: https://rtc-web.linkv.fun/Web_SDK/dist/liveme/OctopusRTC-2.0.0.js
    - liveme 官网:https://www.liveme.com/zh/
    - demo 地址:https://rtc-web.linkv.fun/Web_Demo/videocall/dist/
- 工作职责：
    - 自研了一套基于webrt 的 sdk, 主要功能点就是推流,拉流,结束推流, 结束拉流,开关摄像头, 开关麦和声音,屏幕共享流,视频数据分析等.
- 音视频推流流程(简述):
    - 发送 http 给 rtc 调度中心, 获取socket 链接的 url,同时调度中心获取到客户端 ip , 进行调度.
    - 通过 登录建立 socket 链接.
    - 通过 getUserMedia 获取视频 track
    - 初始化 peerConnection通过 addTrack 将本地 Track 加入, transceiver 初始化
    - negotiationneeded 事件被调用,这这个事件后, 我们可以 createOffer setLocalDescription
    - signalingstatechange 会变成 have-local-offer
    - ice 开始收集 candidate, icegatheringstatechange 事件触发(gathering) (complete)
    - 收到服务端 answerSDP setRemoteDescription 将 sdp注入
    - ice 开始checking ,connectioned
    - connectionstatechange connected,通道建立成功开始推流到服务端
- 链路流程:
    - SDK-Edge-SingleCC(将rtc的流 通过 ffmpeg 转为 rtmp)-cdn(网宿)-观众
- 质量参数:
    - 码率, 帧率, 分辨率, 延迟（Latency）,带宽（Bandwidth)丢包率（Packet Loss),视频质量（Video quality),抖动（Jitter)
    - 288P: 350kbps
    - 480P: 500kbps
    - 720P: 1000kbps
    - 延时：为了保证音视频的实时性，推荐客户端到媒体服务器的 RTT 小于 150ms
    - 丢包：在媒体层面能够抵抗 30% 的随机丢包
- 总结:
    - 断网重新 peerConnection 通道复用, ice 不能重新收集, 通过 createOffer restart 来实现
    - 切换视频源, sender.replaceTrack(videoTrack);

## 2. linkv-octopus-miniprogram(北京乐我无限猎豹移动)

- 描述: 微信小程序端 sdk,基于小程序音视频组件推送和接受 rtmp 流(小程序音视频组件只支持 rtmp 格式流),可以实现多端互通. 应用在豹小秘产品(猎豹移动)
- 技术实现:
    - 技术栈 Webpack,Typescript,EventEmitter,小程序 live-player,live-pusher 和socket.io ,rtmp等
    - 使用 webpack 作为构建工具进行模块化,库类开发, 构建生成 sdk
    - 各模块之间通过 EventEmitter 进行订阅分发事件通信
    - 引入了 ts 做到静态化检查
    - 基于 socket.io 做信令传输
    - 基于小程序 live-pusher 和 live-player 推送 rtmp 流
    - 视频质量通过埋点上报神策日志和大数据,通过神策和水晶球平台进行数据分析

- 地址:
  - npm 下载 npm i linkv-octopus-miniprogram
  - cdn 地址
- 工作职责: 小程序 sdk 项目的开发,问题修复,代码优化.
- 问题:rtmp 延迟比较大 500ms 以上

### 3. 水晶球平台(北京乐我猎豹移动) http://soa.linkv.fun:8181/

- 描述: 音视频数据通过埋点上报到大数据后, 没有直观展示的视频质量的地方,参考竞品厂商, 设计出一套水晶球平台, 主要供后端配置和数据出图分析.
- 技术实现:
  - 技术栈 ant-design-pro 框架(集成了 ant-design, react, react 全家桶)
  - 基于 mock.js 的mock 数据
  - 路由自动生成
- 工作职责: 整个水晶球整个平台的开发.
- 总结: 通过Ant Design Pro，掌握前端工程化开发的流程和方法，熟悉 Ant Design 组件库的使用，提高开发效率.
- 总结: 通过水晶球平台使用 React 有以下总结
  - React 如何实现响应式的呢? React 响应式原理是基于虚拟 DOM 和状态更新机制实现的. 通过虚拟 DOM 的比较和更新, React 可以高效的更新组件的显示. 而通过状态更新机制. React 可以实现组件的响应式
  - React 如何实现父子组件数据传递 在 React 中，父组件向子组件传递数据时，数据是单向流动的，即父组件可以向子组件传递数据，但子组件不能直接修改父组件的数据。如果需要子组件修改父组件的数据，可以通过回调函数的方式来实现。
  - react 的事件机制 React 的事件
  - React 的生命周期 挂载阶段, 更新阶段, 卸载阶段
  - React 的 jsx
  - react hook 的使用
  - react 的有状态组件和无状态组件

### 4. linkv-recorder-sdk(北京乐我无限猎豹移动)

- 描述: linkv-recorder-sdk 是一款 web 端视频录制和截图的(录制完即可播放的)sdk, 可以快速分片上传到, s3 服务端, 实现上传完即可播放和观看缩略图, 已经应用到至录 web 插件和 web 端,pc端.
- 地址: npm 下载 npm i linkv-recorder-sdk
- 技术实现
  - 技术栈: Webpack, Typescript,EventEmitter, socket.io, getUserMedia ,canvas
  - 使用 Webpack 作为构建工具进行模块化开发
  - 引入了 Typescript 做到静态化检查
  - 通过 getUserMedia 获取视频源 通过 MediaRecorder 进行视频录制切片,生成 WebM 格式的切片文件
  - 引入 ebml.js 获取 seekable的视频头信息,计算添加到第一个切片上,实现可以拖拽
  - 通过 canvas 定时截取视频的图片
  - 播放端 web 采用的是 Dplayer 来进行播放
- 总结:
  - 在此进行了技术尝试开始的方案是将 ffmpeg 编译成 wasm, 将 mediaRecorder 获取的切换,通过 ffmpeg 转成 ts, 写 m3u8 文件,问题是浏览器内存会变高,并且写成的 m3u8 文件的文件信息不

对, 效果很差方法.
    - 在生成 WebM 时, 不能通过 MediaRecorder 获取 seekbale 的头信息, 所以通过 ebml 生成视频头, 添加到第一个切换, 实现播放端自由拖拽

## 5. 录制管理中心(北京乐我无限猎豹移动) https://my.video321.net/

- 描述: 录制的管理中心, 实现了录制的文件管理, 团队管理,
- 技术实现
    - ant-design , react, webpack 等
- 主要职责: 站点的部署和文件管理功能模块的开发
- 总结: 因为面向的用户为国外所以考虑到快速访问和负载均衡的问题,这个部署过程是这样的:
    - 域名解析到 lb(负载均衡)服务上
    - https 证书挂载在 lb 服务器上, 因为证书不能反向代理
    - lb 服务器根据规则反向代理到 cdn 服务上
    - cdn 服务根据规则回源到源站上

## 6. ai后台管理系统（北京乐我猎豹移动）http://admin.aiartface.com(目前只能是公司办公网络能访问)

- 描述: ai 项目的前端管理后台实现了上传图片, 用户权限, 菜单管理
- 技术实现
    - 技术栈：Vue.js、Element UI、Vue Router、Axios、ES6、sess
    - 基于开源框架 vue-admin-better
    - 通过路由懒加载,细化组件加载力度
    - 利用路由守卫实现菜单级别的权限管理
    - 基于 mock.js 的数据调试
    - 利用 docker+ nginx 部署前端页面
- 工作职责: 负责整个项目的开发和部署,图片的上传主要上传到 s3 上
- 学习到 Vue 的内容
    - Vue 的数据响应式 数据劫持和订阅分发模式来进行响应式
    - Vue2 数组和对象的响应的区别
    - 双向数据绑定
    - 指令
    - 组件
    - Vue 的模版语法
    - Vue 的路由
    - Vuex
    - 基于 promise 的 axios http 请求
    - 父子组件数据传递

## 7. IMSDK开发(北京乐我无限猎豹移动)

- 描述: 乐我无限开始使用的是融云的IMSDK, 在 后来我们自研了一套IMSDK, 主要运用在 liveme tob端
- 工作职责: 主要是维护此项目, 添加新功能

## 8. pc 端的会议系统(北京乐我无限猎豹移动) 需要看一下 electron api

- 描述: 因为疫情影响,公司缺乏一款自研的视频会议产品, 在这个大环境下, 我们基于 electron-Vue 研发了一套自研的视频会议, 功能类似于飞书
- 技术实现:
  - 技术站: electron-vue 里边包含 electron 和 vue 全家桶等
  - electron 中主进程和渲染进程的区分,每一个窗口都是一个渲染进程
  - electron-log 日志落到本地, 上传到日志服务
  - 通过 electron-builder 进行打包
- 总结:
  - 多窗口之间通过 ipc 通信
  - 在打开其他窗口时,加载很慢,通过窗口队列解决此问题
  - 全屏隐藏窗口, 解决特殊样式比如,工具条 hover 显示的内容, 弹窗中间显示等
- 问题:
  - electron-vue 框架比较古老, 比较笨重,不建议使用
  - 多窗口打开比较慢, 通过窗口队列加载
  - vue 内容本地化, 建议后期可以将前端项目单独部署, 通过 URL 引入,实现热加载

## 9. 前端云控平台(蓝将技术)

- 描述: 一个云控手机的 web 平台, 页面展示多个手机桌面, 实现单击某个云手机或者群控云手机.帮用户实现云手机功能
- 技术实现:
  - Vue全家桶实现页面功能
  - Websocket 跟服务端建立链接,进行手势控制,点击滑动等,同步到 云客户端
  - 获取到 云端 ip , 通过 socket 进行推送 h264 的流
  - 通过 ws-avs 库进行解码播放.
- 总结
  - 画面不清晰, 通过 canvas 画出的画面不清晰,通过缩放解决了此问题
  - 在 canvas 绘制的过程中会出现绿边的问题,视频流的大小不精确导致的
  - 在低端手机上延迟很大, 并且没有做数据丢弃处理.
  - 后期建议技术方法直接使用 webrtc

## 10. 会议模式视频通讯能力库(融云) https://cdn.ronghub.com/RongRTCLib_Web_1.6.3.js

- 描述: 通过输入房间号加入音视频会议的 SDK
- 能力
  - 基本通话：输入房间号加入会议实现音视频
  - 会议控制：基于音视频通话实现会议控制，比如主持人打开或者关闭其他人的音视频
  - 屏幕共享：基于插件获取到屏幕共享流，然后共享桌面或者应用程序
- sdk 的使用 , 目前融云 tob 面向的用户, pc 和 web 端都是使用的是通讯能力 sdk.
- 技术实现
  - 基于 webpack 采用模块化开发,最后打包生成 sdk
  - 采用订阅分发模式处理异步事件
  - 基于 webrtc 技术获取媒体流,和流传输
- 参考文档 tps://www.rongcloud.cn/docs/#rtc
- 项目职责：负责开发和维护 RongRTCLib 库

## 11. 音视频通话通讯能力库(融云)https://cdn.ronghub.com/RongIMLib-2.3.5.min.js

- 描述: 通过打电话的方式来实现音视频通话,开始使用的是声网的,后来自研了一套
- 依赖: ： RongCallLib SDK 依赖于 RongIMLib(融合IMSDK) 和 RongRTCLib(音视频 SDK)，是两者的上层封装
- 功能:
    - 单聊 : 一对一视频聊天
    - 群聊 : 多人音视频
    - 群聊邀请： 群聊中邀请其他人加入
    - 会议控制 ： 群聊中控制他人的音视频
    - 屏幕共享：基于插件获取到屏幕共享流，然后共享桌面或者应用程序
- 技术实现：
    - RongCallLib 的实现采用了函数编程，采用闭包将变量私有化
    - 基于 Underscore 定义了 uti.js
    - 通过发送定义对象消息实现音视频通讯通话

### 12. 企业级通信解决方案 RCE

- 描述: 基于 electron 构建的桌面版应用程序
- SDK 能力
    - 单聊: 一对一视频聊天
    - 群聊: 多人音视频
    - 会议控制: 群聊中控制他人的音视频
    - 屏幕共享: 基于插件获取到屏幕共享流，然后共享桌面或者应用程序
- 技术实现:
    - electron + vue 全家桶
- 项目职责: 主要负责 rce 平台, 音视频模块的开发(最多支持九路视频)

## 教育背景

2010.9 -2013.9 山东滨州学院 专科 信息工程

## 自我评价

在多年的前端开发经验中,我具有基于 Webpack Typescirpt WebRTC 音视频 SDK 和录制等 SDK 开发经验, 具有 Vue 和 React 的项目开发和优化经验, 具有良好的团队意识，具有自我驱动意识, 我乐于学习新的知识和技术,并能将其应用于实际项目中, 我注重代码质量和可维护性,并且能够根据团队需要灵活调整工作内容和方式