2021/7/10 - B 站直播间

前端早早聊大会免费福利专场

早

可视化在数字银行中的应用实践

雅男：Lazada-买家前端专家@阿里 | 14:00

如何从零打造一款可视化大屏编辑器

阿龙：BI 与大数据组负责人@涂鸦 | 15:00

如何从 0 到 1 建设前端性能监控系统

李振：前端监控团队负责人@腾讯云 | 16:00

前

长按扫码报名，进群领取录播/讲稿/PPT

从零开始开发一款可视化大屏编辑器

PS：个人学习型项目 ➔ 接地气

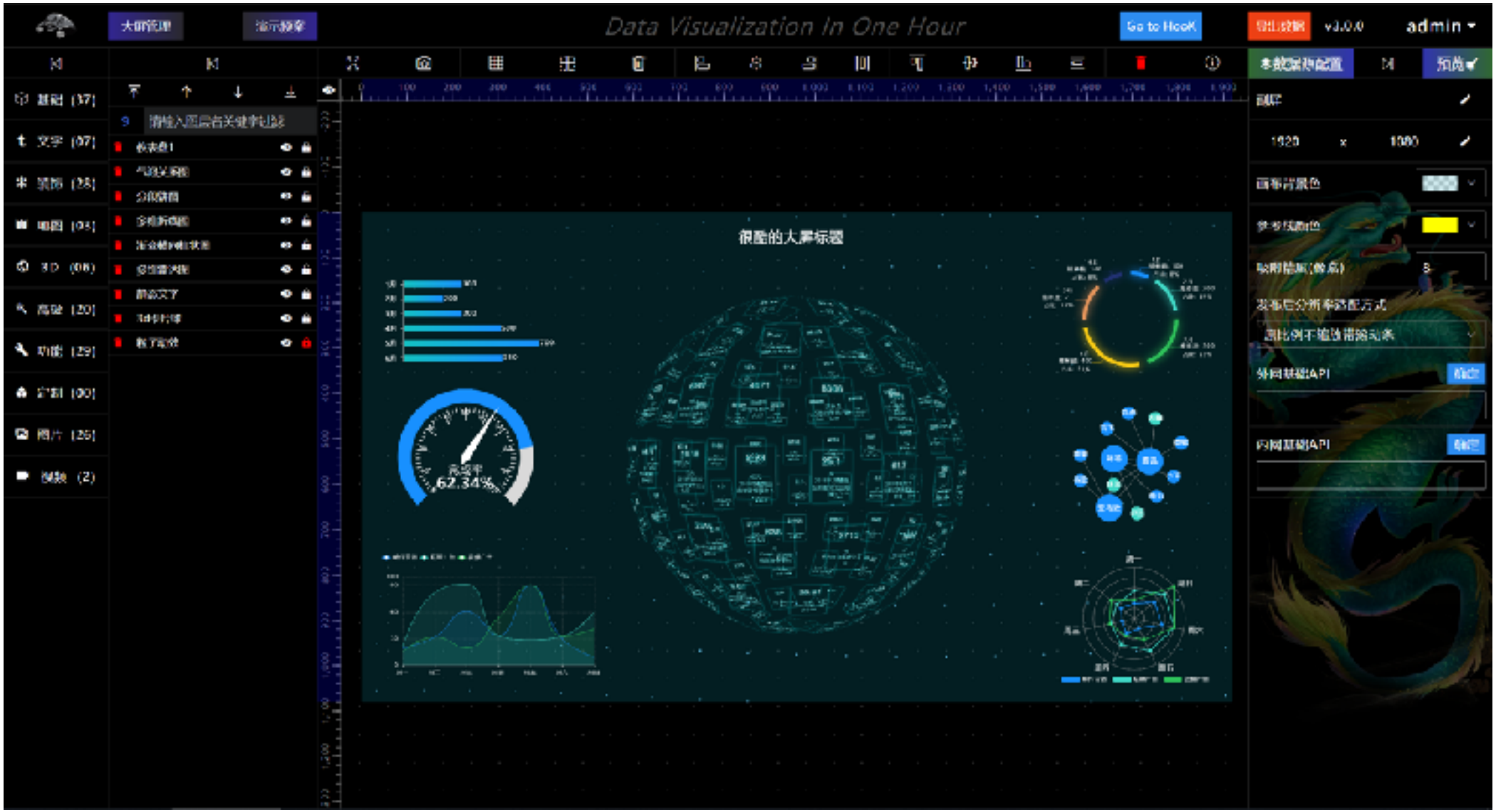
阿龙 2021.7.10

目录

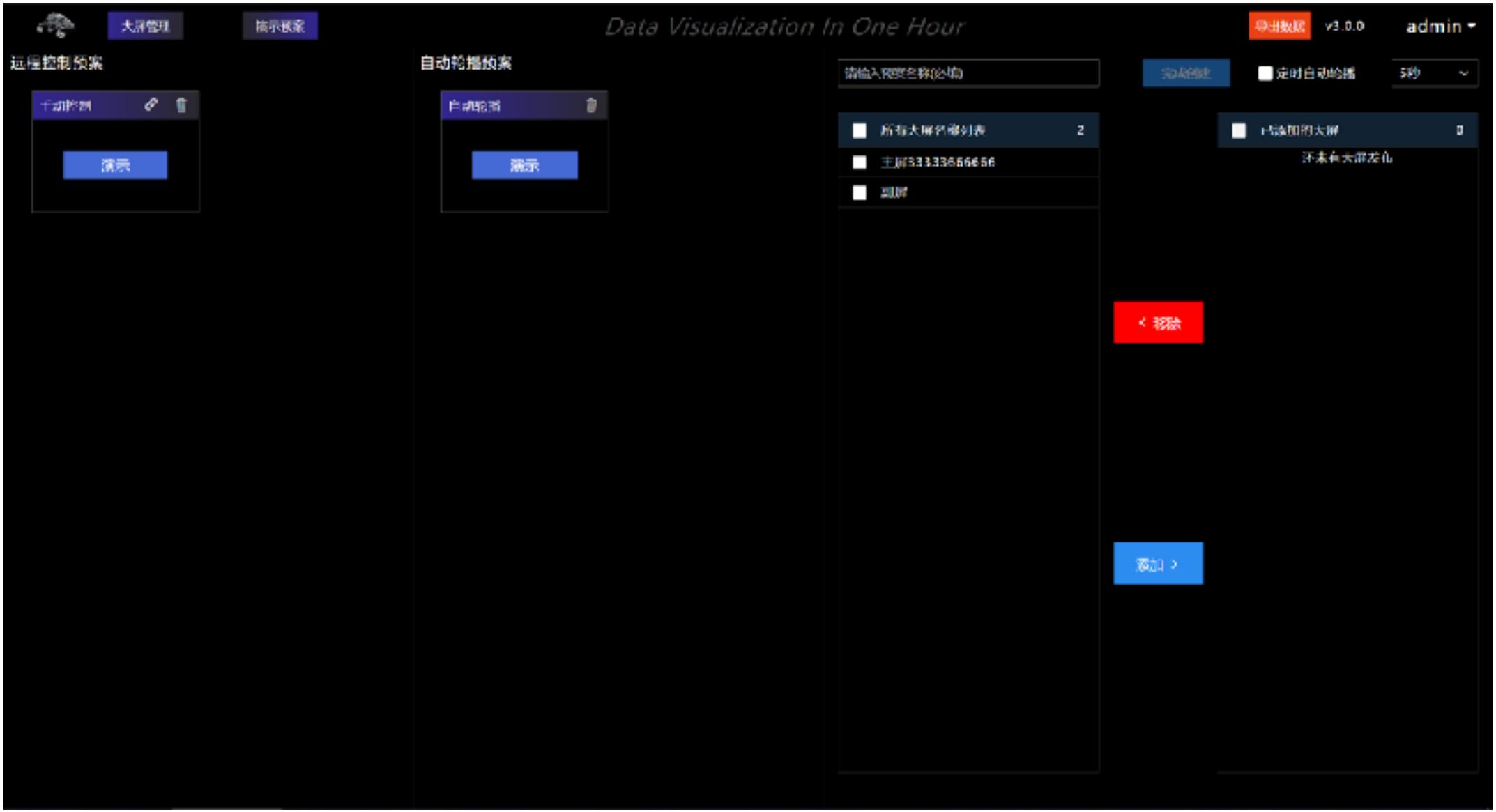
1、成果展示-大屏管理页


Data Visualization In One Hour
v3.0.0
admin
创建模板
+新建分组
1920 x 1080
2021-02-17 12:51:16
已发布
1920 x 1080
2021-03-05 20:38:55
已发布
1920 x 1080
2021-07-07 19:39:00
主屏
副屏

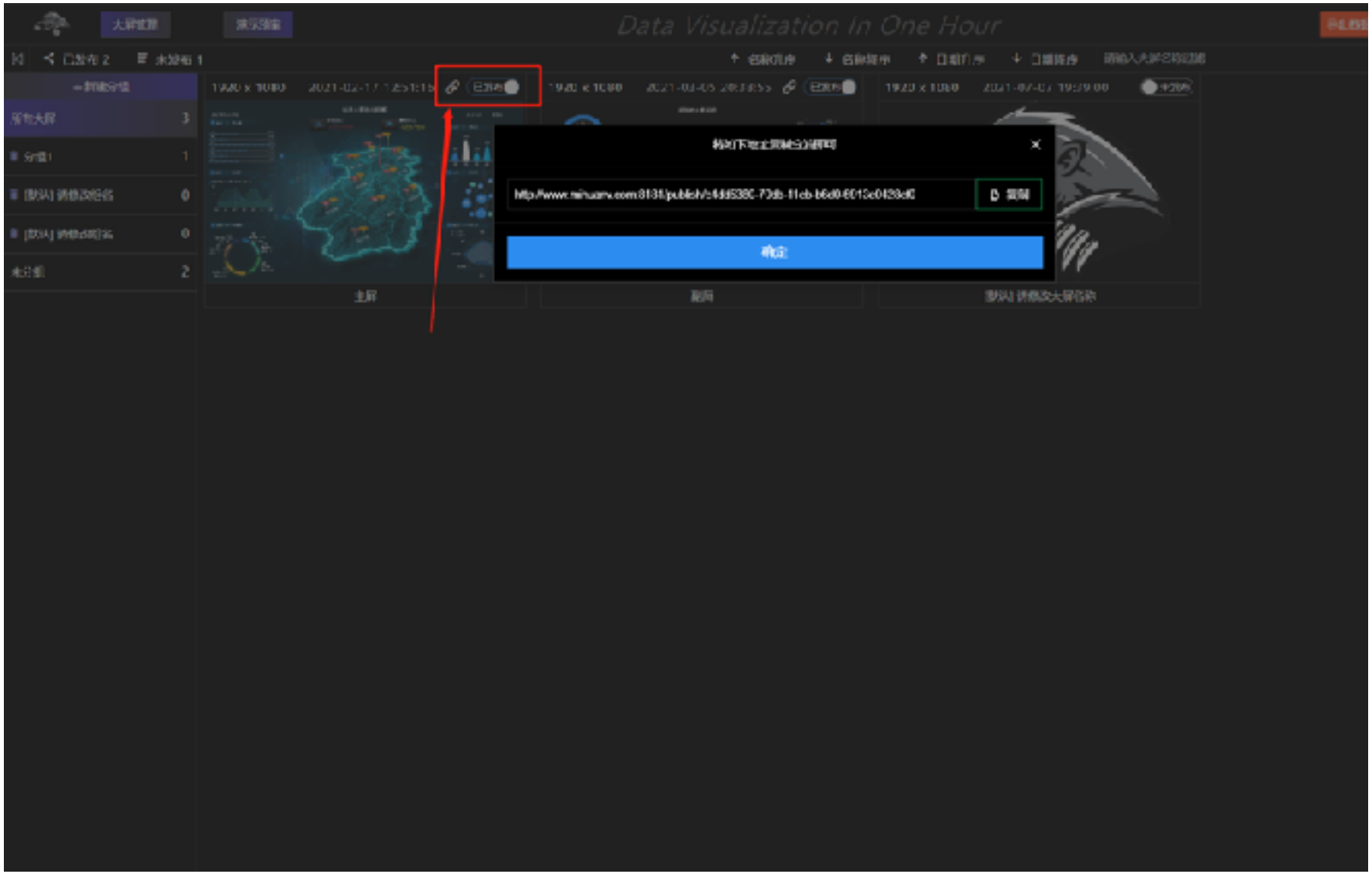
1、成果展示-大屏编辑页



1、成果展示-演示预案管理页

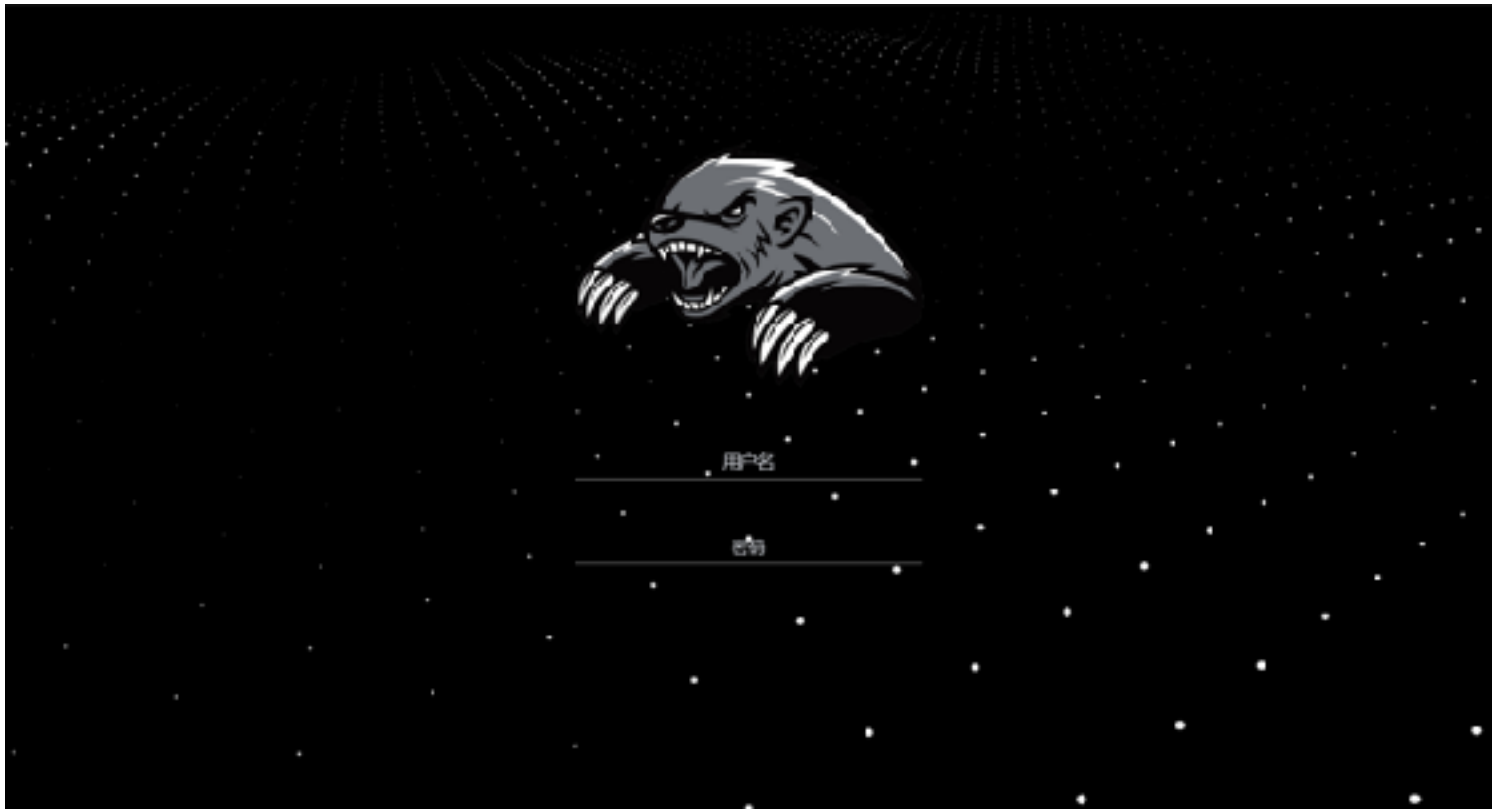


1、成果展示-发布大屏



2、功能介绍

生死看淡 不服就干 http://www.mihuanv.com:8181



(1) 内置丰富的组件
(2) eCharts / highCharts
(3) 组件数据交互
(4) 页面全局hook
(5) 丰富的舞台编辑功能
(6) 数据整形
(7) 组件移动过程中吸附
(8) 入场动画
(9) 。。。

感兴趣的可以联系开账号体验

3、为什么做

1、降本提效（百分之七八十的大屏基于工具开发交付）

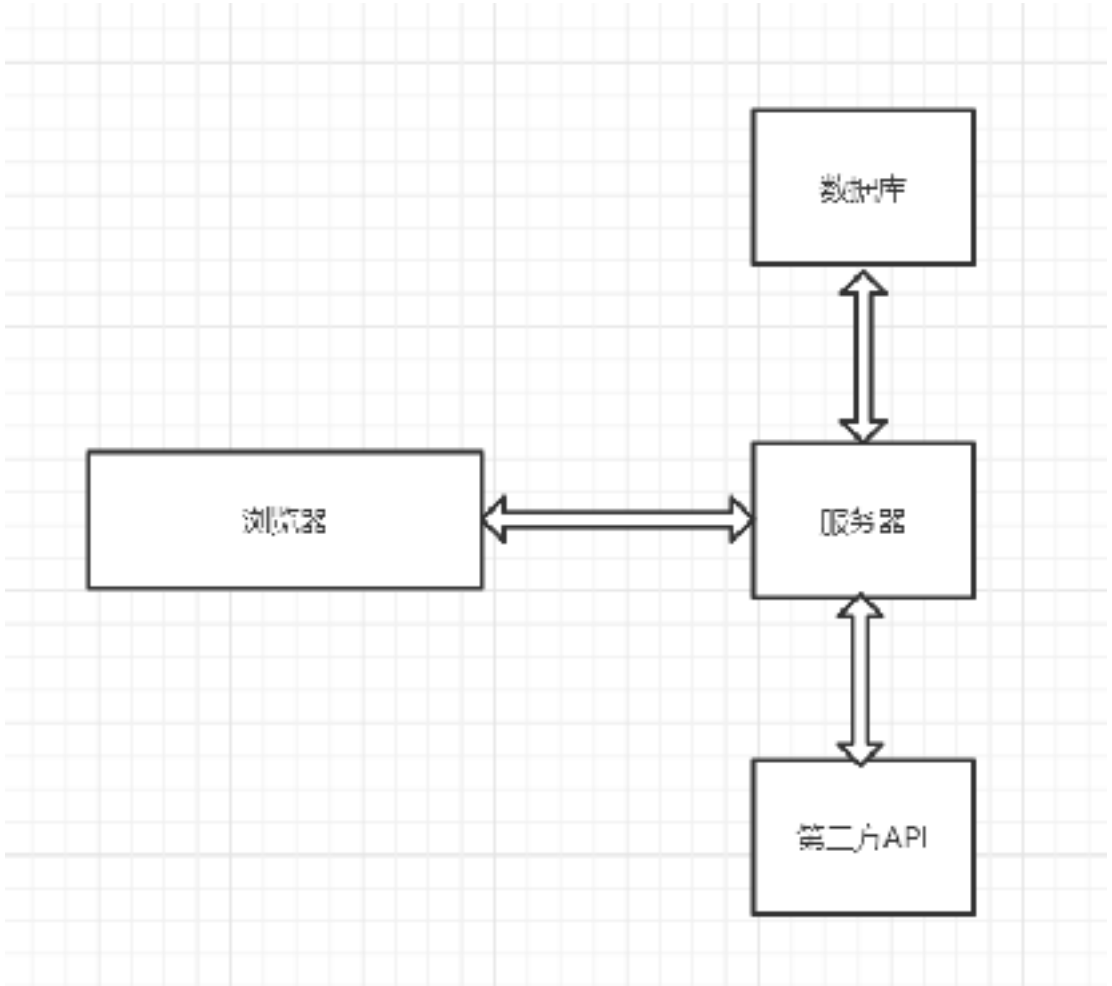
2、**个人项目，能力，展示（学习前端+后端+数据库+部署运维→独立完整项目的开发）**

3、组件汇总积累的平台、市场

4、大屏模板积累的平台、市场

5、构建自己的生态（交友）

4、怎么做➔技术选型



（1）前端

框架：vue / react / angular / next.js / umi.js / nuxt.js 等

UI 组件库：antD / iView / elementUI 等

用户登录：jwt

（2）后端

node.js java 等

express / koa / egg.js / nest.js / spring boot 等

（3）数据库

MySQL / PgSQL / MongoDB / json-server 等

（4）部署运维

node.js（pm2） / nginx / docker（k8s） 等

4、怎么做➔ package.json 介绍

1、package.json 介绍

2、文件目录介绍

3、前端入口及主要设计思想介绍

4、组件设计及配置方式介绍

5、演示预案方案

6、后端入口及主要设计思想介绍

7、数据库相关介绍

8、部署运维相关介绍

4、怎么做➔ package.json 介绍

vue

view-design

lodash-move

dom-to-image

express

busboy

ws

json-server

axios

jsonwebtoken

@tweenjs/tween.js

video.js

three

textures

maptalks.three

maptalks

highcharts

d3

echarts

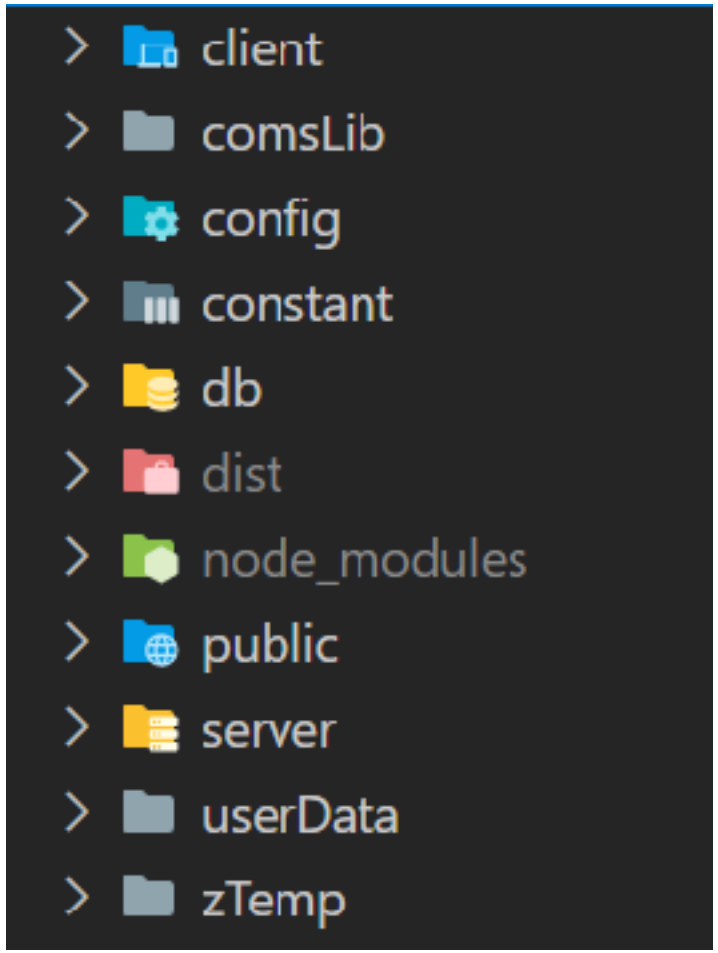
coordtransform

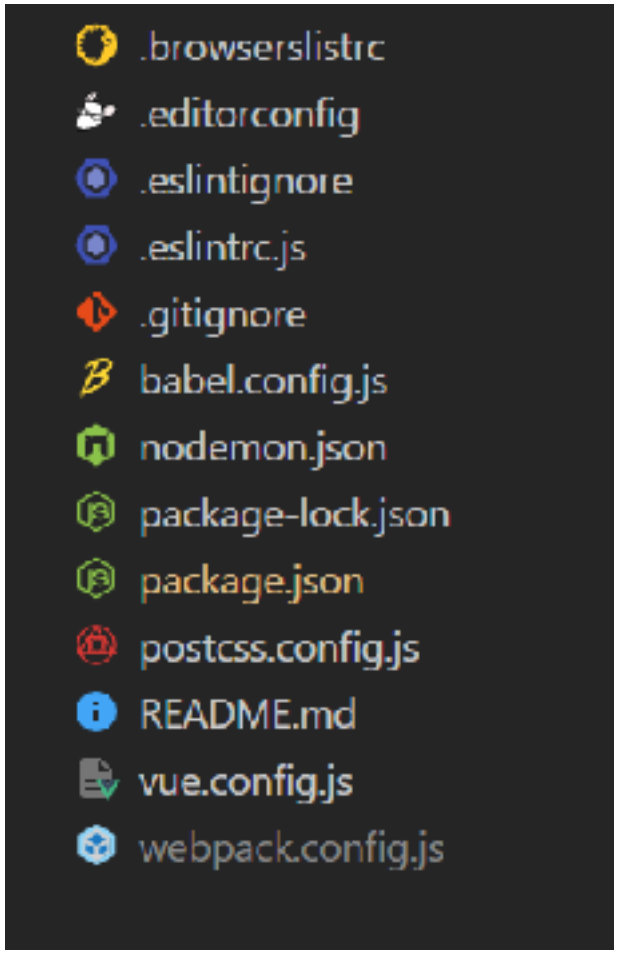
cesium

@turf/turf

xgplayer

4、怎么做→文件目录介绍


client
comsLib
config
constant
db
dist
node_modules
public
server
userData
zTemp


.browserslistrc
.editorconfig
.eslintignore
.eslintrc.js
.gitignore
babel.config.js
nodemon.json
package-lock.json
package.json
postcss.config.js
README.md
vue.config.js
webpack.config.js

4、怎么做→前端入口及主要设计思想介绍

```
const systemInitial = async () => {
  try {
    const isServerConfigSuccess = await getServerConfig();
    if (isServerConfigSuccess) {
      globalRegisterComs();
      registerConfigComs();
      createWebSocket();
      renderFn(200);
    } else {
      console.log('从后台获取服务器配置相关参数错误');
      renderFn(500);
    }
  } catch (err) {
    console.log('系统初始化错误:==', err);
    renderFn(500);
  }
};

systemInitial();
```

```
const renderFn = (code) => {
  new Vue({
    router,
    store,
    render: (h) => {
      switch (code) {
        case 200:
          return h(App);
        case 500:
          return h(ErrorPage);
        default:
          return h(ErrorPage);
      }
    },
  }).$mount('#rootAppId');
};
```

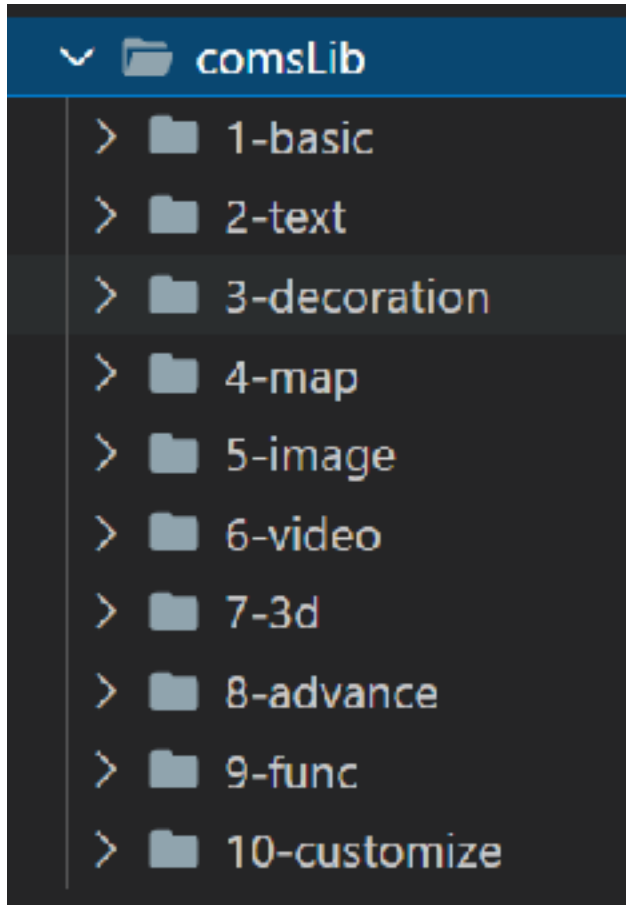
4、怎么做→组件设计介绍→全局注册组件

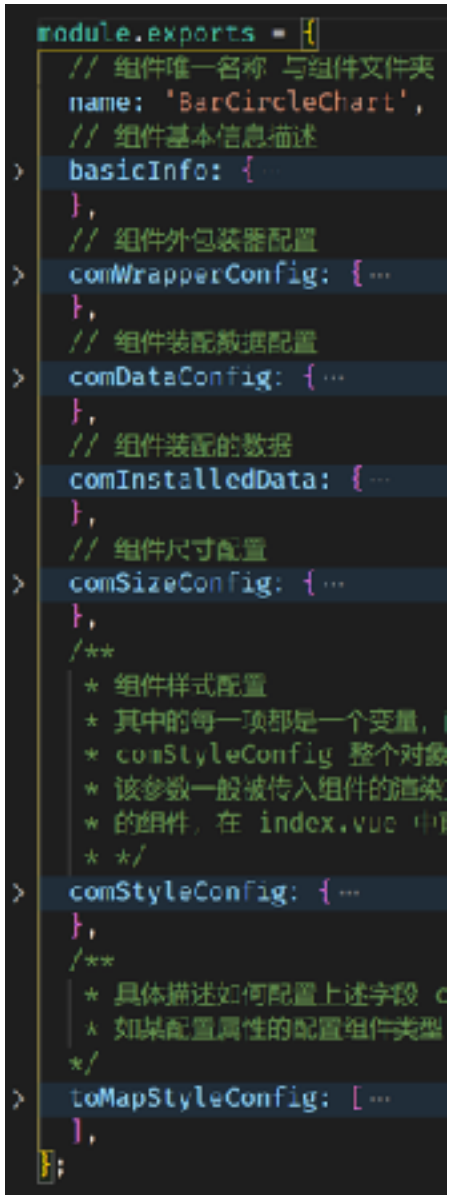
```
const globalRegisterComs = () => {
  try {
    const requireComponent = require.context(
      '../comsLib',
      true,
      /index.vue/,
    );
    requireComponent.keys().forEach((fileName) => {
      const componentConfig = requireComponent(fileName);
      Vue.component(
        componentConfig.default.name,
        componentConfig.default,
      );
    });
  } catch (err) {
    console.log('globalRegisterComs error:==', err);
  }
};
```

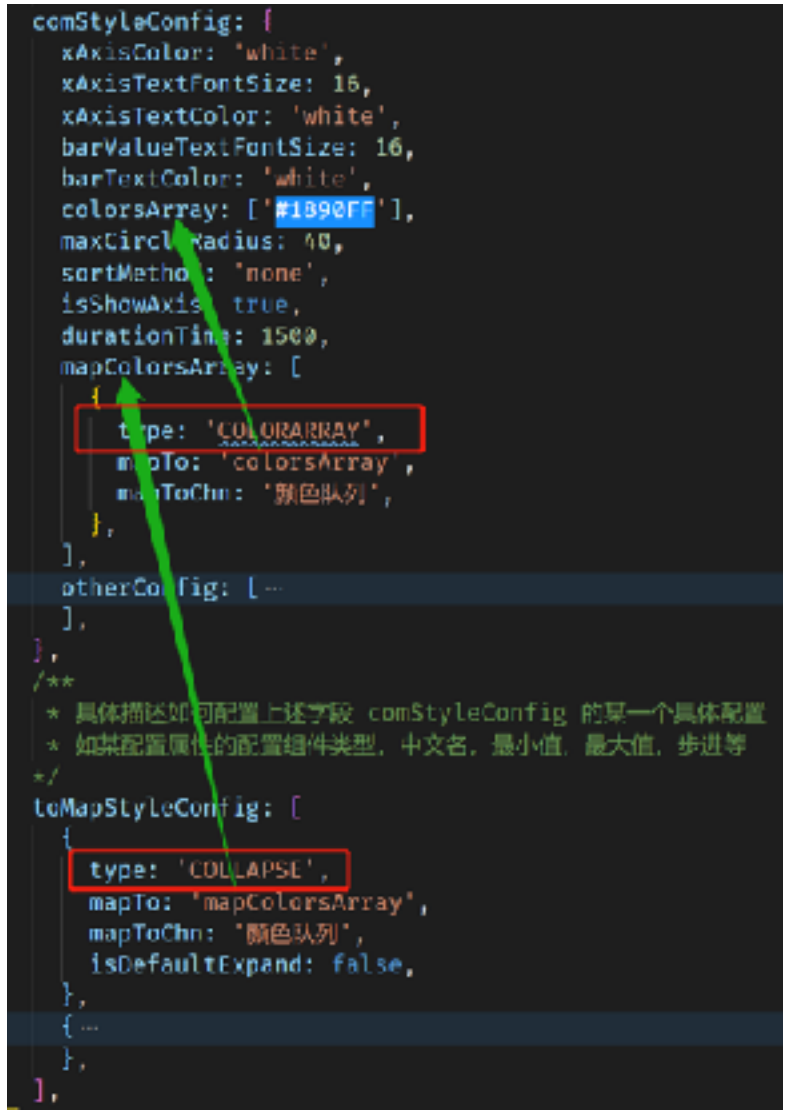
4、怎么做→组件设计介绍→组件名的动态组件

```
<ComWrapper v-for="comPackageJson of onStageComsList"
  :propComPackageJson="comPackageJson"
  @on-right-click="onRightClick"
  @show-x-helper-line="onShowXHelperLine"
  @show-y-helper-line="onShowYHelperLine">
  <component slot="comSlot" :is="comPackageJson.name"
      width: `${comPackageJson.comSizeConfig.width}px`
      height: `${comPackageJson.comSizeConfig.height}p
      'pointer-events': $route.name !== 'editor' ? 'au
    }"></component>
</ComWrapper>
```

4、怎么做→组件配置方式介绍

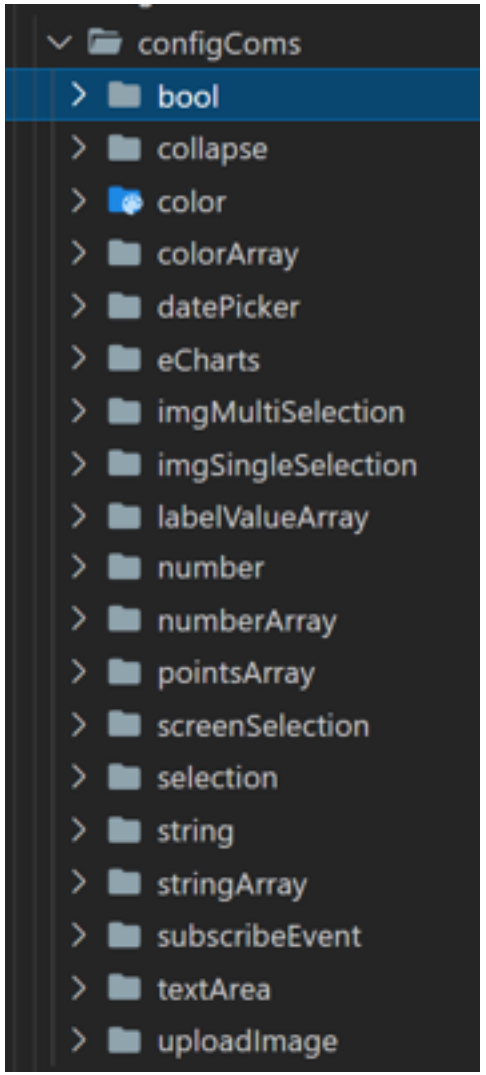




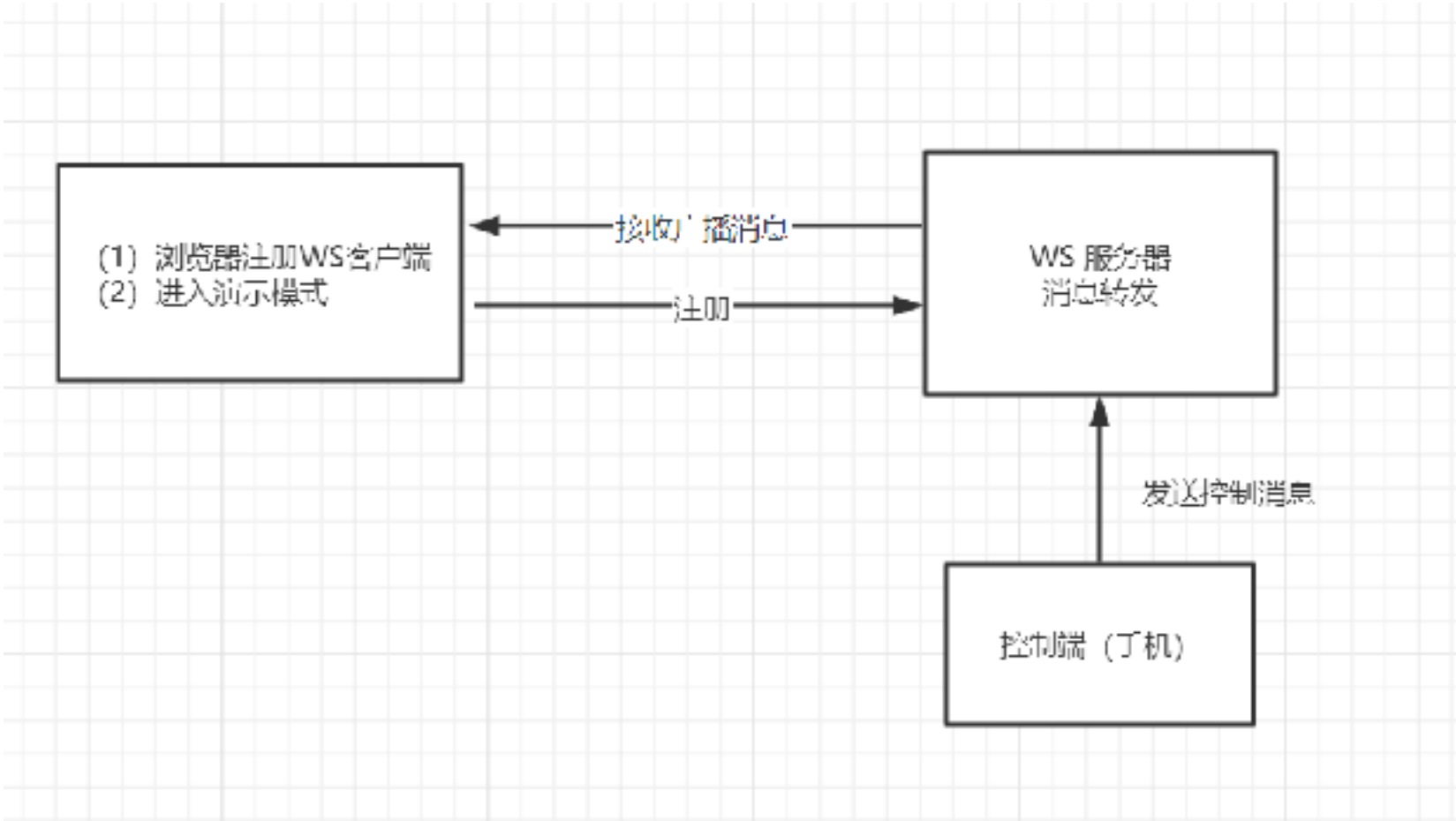


4、怎么做→组件配置方式介绍

```
const registerConfigComs = () => {
  try {
    const requireComponent = require.context(
      './pages/editor/right/configComs',
      true,
      /index.vue/,
    );
    requireComponent.keys().forEach((fileName) => {
      const componentConfig = requireComponent(fileName);
      Vue.component(
        componentConfig.default.name,
        componentConfig.default,
      );
    });
  } catch (err) {
    console.log('registerConfigComs error:==', err);
  }
};
```



4、怎么做→演示预案方案→逻辑框图



4、怎么做→演示预案方案→前后端代码

```
module.exports = (httpServer) => {
  /* ================== create websocket server ==============
  const wss = new WebSocket.Server({ server: httpServer });
  if (wss) {
    console.log('websocket server created success\n');
  } else {
    console.log('websocket server created failed\n');
  }
  function noop() {}

  function heartbeat() {
    // console.log('===========on ping-pong==========');
    this.isAlive = true;
  }

  /* ==================== on connection ======================
  wss.on('connection', (ws) => {…
  });
  /* ==================== patrol to detect and close broken c
  let pingTimeoutId = null;
  function ping() {…
  }
  ping();
};
```

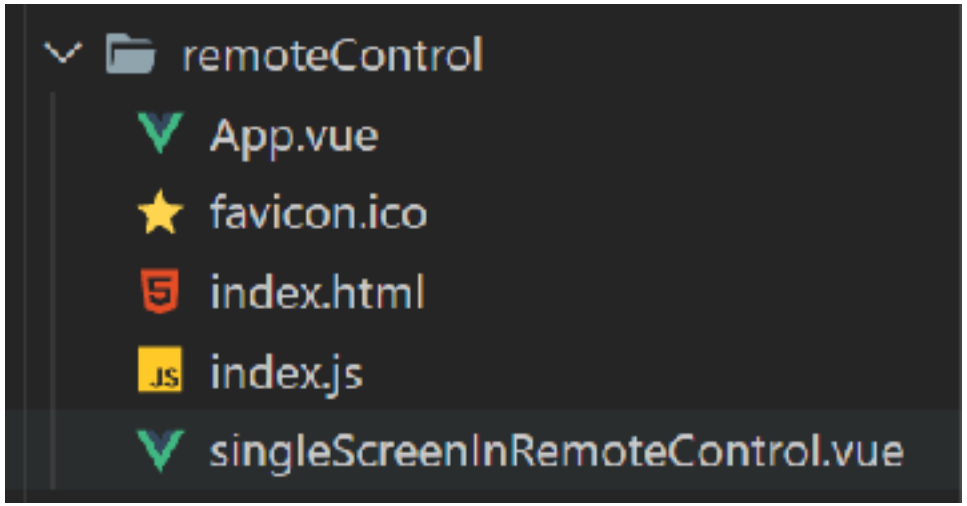
```
const createWebSocket = () => {
  const wsUrl = `${window.WS_PROTOCOL}://${window.
  window.ws = new WebSocket(wsUrl);
  window.ws.onopen = () => {
    console.log('app.js window.ws on open');
    window.wsConnected = true;
  };

  window.ws.onerror = (err) => {
    console.log('window.ws client onerror', err);
  };

  window.ws.onclose = () => {
    console.log('window.ws client onclose');
  };
};
```

4、怎么做→控制端MPA介绍

```
module.exports = {
  productionSourceMap: isDev,
  // publicPath: isDev ? '/' : '/assets/',
  pages: {
    remote: {
      entry: './client/pages/remoteControl/index.js',
      template: './client/pages/remoteControl/index.html',
      filename: 'remote.html',
      chunks: ['chunk-vendors', 'chunk-common', 'remote'],
    },
    app: {
      entry: './client/app.js',
      template: './public/index.html',
      filename: 'index.html',
      chunks: ['chunk-vendors', 'chunk-common', 'app'],
    },
  },
```



4、怎么做→后端入口及主要设计思想介绍

```
const fs = require('fs');
const path = require('path');
const http = require('http');

const express = require('express');
const bodyParser = require('body-parser');
const compression = require('compression');
const cookieParser = require('cookie-parser');

// const jwt = require('jsonwebtoken');···

const apiHandler = require('./api/handle-api.js');

const {···
} = require('../config/index.js');

const app = express();
app.use(cookieParser());
app.use(compression());
app.use(bodyParser.json({ limit: '50mb' }));

const HTTP_SERVER = http.createServer(app);
```

```
function setNoCache(req, res, next) {···
}

app.use('/static', express.static(path.join(__
app.use('/comsLib', express.static(path.join(_
app.use('/userData', express.static(path.join(
}));
app.use('/api',···
);

/* production mode */
if (!isDev) {···
}

app.use((err, req, res, next) => {···
});

HTTP_SERVER.listen(ECS_PORT, () => {···
});
```

4、怎么做→ API组织及组件库读取

```
const router = require('express').Router();

const createWSS = require('../wss/wss.js');
const UserApi = require('./userApi.js');
const CbsApi = require('./cbsApi.js');
const PublicApi = require('./publicApi.js');
const DataSourceApi = require('./dataSourceApi.js');
const DataQuotaApi = require('./dataQuotaApi.js');
const RemoteControlApi = require('./remoteControlApi.js');

const CusApi = require('./cusApi.js');

module.exports = (httpServer) => {
  createWSS(httpServer);
  UserApi(router);
  CbsApi(router);
  PublicApi(router);
  DataSourceApi(router);
  DataQuotaApi(router);
  RemoteControlApi(router);
  CusApi(router);

  return router;
};
```

```
// 从本地文件夹获取所有图片资源 不涉及数据库操作
router.get('/cbs/getAllComsList', (req, res) => {
  try {
    fs.readdir(comsRootPath, (err1, firstDirsArray) => {
      const firstDirsArrayWithoutDotFile = firstDirsArray
      // console.log('firstDirsArrayWithoutDotFile:--', f
      firstDirsArrayWithoutDotFile.sort((a, b) => {
        const numA = parseInt(a, 10);
        const numB = parseInt(b, 10);
        if (numA > numB) {
          return 1;
        }
        return -1;
      });
      const comsListObj = {};
      firstDirsArrayWithoutDotFile.forEach((firstDirName)
        comsListObj[firstDirName] = [];
        const firstLevelPath = path.join(comsRootPath, fir
        const firstDirNameArray = fs.readdirSync(firstLeve
        // console.log('firstDirNameArray:--', firstDirNa
        const firstDirNameArrayWithoutDotFile = firstDirN
        firstDirNameArrayWithoutDotFile.forEach((comName)
          const comNamePath = path.join(firstLevelPath, co
          const fileContent = require(comNamePath);
          comsListObj[firstDirName].push(fileContent);
        });
      });
      res.json({
        success: true,
        data: comsListObj,
      });
    });
  } catch (err) {
    console.log('getAllComsList err:--', err);
    res.json({
      success: false,
      data: 'getAllComsList error',
    });
  }
});
```

4、怎么做→数据库服务

```
const ip = require('ip');
const path = require('path');
const jsonServer = require('json-server');
const { DB_PORT, isDev } = require('../config/index.js');

const server = jsonServer.create();
const router = jsonServer.router(path.join(__dirname, 'db.json'));
const defaultMiddleware = jsonServer.defaults({
  // static: '', // path to static files
  logger: isDev, // enable logger middleware (default: true)
  // bodyParser: true, // enable body-parser middleware (default: true)
  noCors: true, // disable CORS (default: false)
  // readOnly: false, // accept only GET requests (default: false)
});

server.use(defaultMiddleware);
server.use(router);
server.listen(DB_PORT, () => {
  console.log(`JSON Server is running at http://${ip.address()}:${DB_PORT}`);
});
```

4、怎么做→部署及运维相关（node.js-http server → pm2 管理）

```
"scripts": {
  "db:dev": "cross-env NODE_ENV=development nodemon db/index.js",
  "dev:client": "cross-env NODE_ENV=development node --max_old_space_size=4096 node_modules/@v
  "dev:server": "cross-env NODE_ENV=development nodemon server/index.js",
  "lint": "vue-cli-service lint --fix",
  "build": "cross-env NODE_ENV=production node --max_old_space_size=4096 node_modules/@vue/cli
  "db": "cross-env NODE_ENV=production node db/index.js",
  "db:deploy": "cross-env NODE_ENV=production pm2 start db/index.js --name 'mihuanv-v3-db'",
  "server": "cross-env NODE_ENV=production node server/index.js",
  "deploy": "cross-env NODE_ENV=production pm2 start server/index.js --name 'mihuanv-v3'"
},
```

```
/* production mode */
if (!isDev) {
  app.use('/', express.static(path.join(__dirname, '../dist')));
  const indexHtml = fs.readFileSync(path.join(__dirname, '../dist/index.html'), 'utf-8');
  app.use('*', (req, res) => {
    res.append('X-Frame-Options', 'DENY');
    res.send(indexHtml);
  });
}
```

5、该项目如何应用?

（1）独立售卖工具（私有化部署，license，时间，组件，模板，功能）？

➔ 终端用户会使用吗？目标用户是不是还是需要开发者?

类似客户想要一个做好的胶片，你总不能直接给他一个 PowerPoint 跟他说用这个

工具可以作出各种你想过要的效果

（2）作为内部大屏开发工具提效？➔我们作为前端开发者可以推荐给自己的公司

（3）➔ 结合业务

5、该项目如何商业应用 ➔　涂鸦 IoT 云开发平台为例

该项目预计将于8月开启公测，欢迎大家添加小助手，可进群报名公测

6、IoT云开发平台——万亿智能物联新赛道

涂鸦智能（NYSE: TUYA）是全球领先的 IoT 云平台, 截至2021年3月31日, 涂鸦 IoT 开发平台累计有超过32.4万注册开发者, 日语音 AI 交互超1.22亿次, 每日设备请求次数840亿次, Powered by Tuya赋能超31万设备SKUs, 产品和服务覆盖超过220个国家和地区。

用户有自己的数据在 IoT 平台上，基于在平台上的数据，结合该工具，可以无缝的使用自己的数据制作大屏，为自己的业务服务

7、欢迎加入涂鸦开发者生态，共建万物互联新世界

参与涂鸦开发者活动

https://promotion.tuya.com/develop/developer2

THANKS

Thanks.
世界很大 一起涂鸦.

Question & Answer

第二十九届前端早早聊大会

掘金

数据可视化 | 时空可视化 | 大屏 | 搭建 | 画布 | ☆弯道超车 | 7月17日全天直播

2021 全年行程

1/9 自由职业/副业
1/23 前端团队管理
2/6 小程序|组件化
2/27 页面搭建专场
3/20 大厂招聘面试
4/10 前端搞CI/CD
4/24 前端玩转算法
5/09 前端述职晋升
5/15 前端玩转互动
6/5 跨端搞 Flutter
6/12 编译彩蛋专场
6/26 前端 WebGL
7/17 前端搞可视化
7/24 前端玩转 BFF
8/14 前端 Node.js
8/28 前端安全专场
9/11 Serverless
9/25 前端测试专场
10/16 前端搞监控
11/20 WebAssembly

9:00	《前端工程师的可视化修炼之路》	庞凤	贝壳找房	「基础大前端」负责人
9:50	《如何从零搭建全栈可视化大屏编辑器》	徐小夕	自由职业	「H5-Dooring」开源作者
10:40	《如何打造沉浸式的时空可视化 Web 应用》	刘茜	奇安信	「奇安信雷尔平台」核心开发
11:30	《如何从原理层面上手 Three.js》	木的树	自由职业	「可视化」技术专家
13:00	《地理可视化，不止是炫酷》	芋头	预策科技	技术总监
13:40	《如何设计与实现 L7 地理可视化引擎》	正学	蚂蚁集团	「地理可视化引擎」研发负责人
14:30	《如何设计与实现思维导图可视化方案》	泽辉	小米	「体验效能」前端可视化方向
15:20	《如何构思和开发开箱即用的图表库 - G2Plot》	新茗	蚂蚁集团	「AntV」核心开发
16:10	《如何打造超大规模图可视化画布》	逸达	阿里巴巴	「图可视化引擎」研发负责人
17:00	《如何在交通领域玩转数字仿真与可视化》	婉一	阿里云	「DataV」技术专家
18:00	《如何打造可落地的数字孪生可视化引擎》	刘学	数字冰雹	「可视化引擎」 技术负责人

前端早早聊大会直播

2020 PK 2021

已举办 16 期 100 场

1/11 前端转管理	6/20 前端跨端跨栈
2/29 前端搞基建	6/27 前端女生专场
3/28 前端搞搭建	7/18 前端搞可视化
4/11 前端搞规划	8/15 前端搞构建
4/25 前端搞监控	8/29 前端成长晋升
5/16 Serverless	9/26 前端搞报表
5/30 前端搞微前端	10/17 前端搞组件
5/31 前端搞面试	11/21 前端搞框架
6/13 前端搞文档	12/26 前端搞性能

计划举办至少 20 期 140 场

1/10 前端搞副业	5/29 工程化/Flutter
1/23 前端搞管理	6/05 跨端 Flutter
2/06 前端搞小程序	6/19 福利专场
2/27 可视化搭建	6/26 前端搞 WebGL
3/06 前端搞搭建	6/27 前端搞微前端
3/20 前端搞面试	7/17 前端搞可视化
3/27 菜鸟大前端	7/24 前端搞 BFF
4/10 CI/CD	8/28 前端搞安全
4/24 前端搞算法	9/11 Serverless
5/09 前端搞述职	9/25 前端搞 IoT
5/15 前端搞互动	10/16 前端搞监控
NEW! 11/20 前端搞 IDE	NEW! 12/11 玩转 Node.js

(以实际举办为准，行程/话题/场次会做动态调整)

一招鲜走天下 组合拳闯四海
单主题多讲师 听得懂抄得走

前端早早聊大会

2021年票

解锁 2021 年 140 场干货技术直播

单场大会用户	年票 VIP 用户
平均每期 77 元	平均每期 25 元
平均每场 12 元	平均每场 5 元
-	不限次数发布招聘
-	获得优质简历模板
-	Scott 简历指导及内推
-	2022 年票 <= 7 折
-	其他神秘福利...

￥660

每个月有直播
每一场有录播