# Vue项目遇到的问题和解决方案或知识

1. 登录成功的token如何管理

2. 登录拦截如何做

3. token过期如何处理

4. 左侧菜单如何生成的？

5. 路由如何划分模块的？如何使用的？

6. 项目中跨域如何解决的，原理是什么？

7. 环境变量配置文件的作用什么？举例说明

8. 数组转树如何实现？

9. Excel的上传和下载如何实现？

10. Vue.use()的原理是什么？能做什么？

11. v-model的原理是什么？组件上用v-model如何做？

## 递归实现动态菜单

前面我们一直在说一级、二级、三级菜单的实现逻辑都是相同的：循环子路由，在循环的时候判断子路由是否包含子菜单，如果包含则当前菜单使用el-submenu实现，否则当前菜单使用el-menu-item实现。那这样的逻辑最适合的就是使用递归去实现。

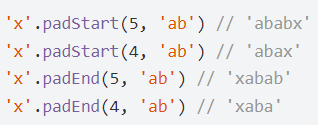
所以我们需要将这部分共同的逻辑抽离出来作为一个独立的组件，然后递归的调用这个组件。

## 插件里面能做啥

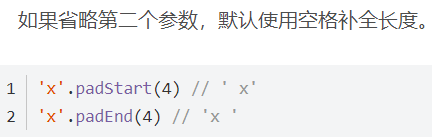
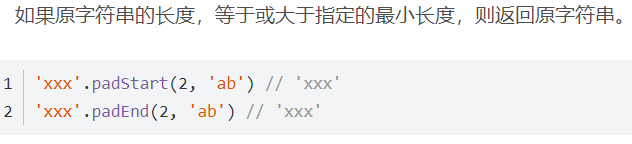


## padStart(),padEnd()方法

ES2017 引入了[字符串](https://so.csdn.net/so/search?q=%E5%AD%97%E7%AC%A6%E4%B8%B2&spm=1001.2101.3001.7020" \t "_blank)补全长度的功能。如果某个字符串不够指定长度，会在头部或尾部补全。padStart()用于头部补全，padEnd()用于尾部补全。



上面代码中，padStart()和padEnd()一共接受两个参数，第一个参数用来指定字符串的最小长度，第二个参数是用来补全的字符串。



## 想要修改网页的页签标题

可以在路由前置守卫的判断token之前就给document.title添加动态的值 比如：document.title= to.meta.title

## 让el-table 表格的序号递增的办法

第一种

<el-table-column type="index" :index='(index)=>{return (index+1) + (page-1)\* size}' fixed="left" label="序号" width="100" align="center">

第二种

  <el-table-column type="index" :index='indexMethod' fixed="left" label="序号" width="100" align="center">

  indexMethod(index) {

      let curpage = this.page; //单前页码，具体看组件取值

      let limitpage = this.size; //每页条数，具体是组件取值

      return index + 1 + (curpage - 1) \* limitpage;

    },

## el-tree组件中返回已勾选的节点的方法

this.$refs.tree.getCheckedKeys()

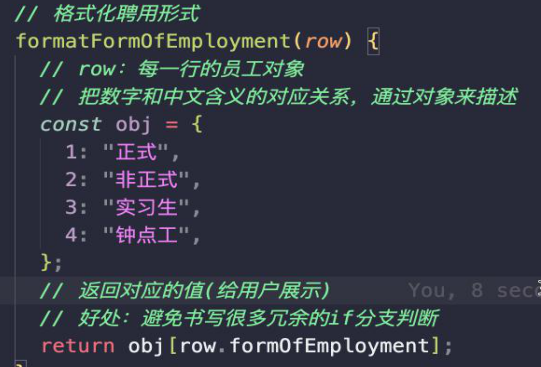
## 项目优化写法

做项目时后台返回的一些数据是以数字做值的，这些数字都有对应得含义

那么当我们想要进行格式化处理时，一般不使用if语句，太low

设置一个对象，来表示这个数字和实际含义的映射关系

比如下图：



当然上图的写法不是很完美，因为这些数据可能是复用,并且它是一个常量，不会变化

所以将其导入一个js文件中，需要使用的时候，引入到对应的组件中使用

完整的写法：D:\前端\Project\Vue\_project\iHRM\src\views\employees\index.vue

搜索format函数

补充一点：在el-table得el-table-column上有一个属性formatter是用来格式化数据的

router/index.js的new Router({ mode:’history’, base:”api/” // 配制的基本路径前面再加上api})

比如：<http://xxxx.xxx/xxx> 🡺 <http://xxxx.xxx/api/xxx>

## 全屏方法

document.documentElement.requestFullscreen()

退出全屏方法

document.exitFullscreen()

在设置命名空间的情况下，vuex模块之间的通信如下：

添加第三个参数，表示这是在根模块下调用的

    commit("permission/setRoutes", [], { root: true });

## 可以通过router.addRoutes() 动态的添加路由

## 新增和编辑的弹出框都是差不多的，所以一般共用一个组件

此时需要区别提交的请求是新增还是编辑，组件的标题是新增还是编辑

一般来说都是以id来区别，新增的数据对象中没有id，而编辑有

## 有些时候当我们点击某个按钮，然后弹出窗口时，有些数据需要渲染到弹出的窗口上,但是网络请求可能有延时，导致窗口出现一会后，数据才获取并渲染到页面上.

此时可以通过await 使得数据获取到之后，再打开弹窗

## 因为props是异步获取的，所以从父组件传递给子组件的自定义属性,子组件不一定都能即使获取到，所以这时候一般都是采用

子组件写好一个方法

父组件通过this.$refs来调用该子组件上的方法，从而将数据传递给子组件使用

## 项目中上传的图片一般都是放在云服务器上，数据库只存储链接

查看md文档

D:\前端\Project\Vue\_project\iHRM\md\11 上传图片.md

C:\Users\86180\Desktop\company\20220822资料\人资项目11天资料\02-笔记\07-员工管理.md 三分之二位置

## 修改第三方组件的类名时，切记切记要加深度选择器

## axios的参数别写错了 method url data

## 通过类名找标签

给标签添加一个类名，然后在浏览器上通过类名查找该标签

## 项目相关知识

vuecli脚手架环境，基于node+webpack来支撑我们写vue项目

默认入口文件main.js->所有代码都要和main.js产生直接/间接的引入关系

打包过程：

执行打包的时候webpack会根据入口的引入关系来构建代码地图，把相关的代码用加载器/插件，翻译处理，输出到js文件中，插入index.html中运行

## 报错信息：closeToken 未定义的这么一个错误

因为响应拦截器配置了，但是没有返回数据，所以报错了

## [require.context实现前端工程化](https://www.cnblogs.com/qiaozhiming123/p/15791783.html)

[require.context实现前端工程化 - 前端乔 - 博客园 (cnblogs.com)](https://www.cnblogs.com/qiaozhiming123/p/15791783.html)

## this.$router.options.routes

返回的是路由表中routes属性的值，也就是路由规则表的值

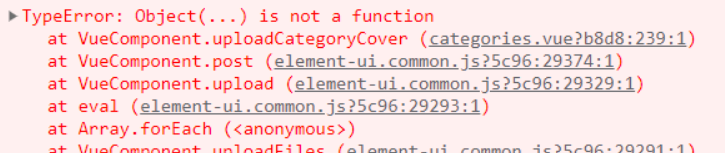
## this.$route如果在js文件中改如何获取?

以下代码可以代替this.$route

最好自己再次打印一下router.currentRoute看看参数

router.currentRoute.path

## 解决 Vue 中报错：TypeError: Object(...) is not a function



## 问题原因：

1 从其他模块导入方法时，引入的方式出错

2 方法重名

## Vuex导出(export default)的对象不能在最上方

export default {

state,

mutations,

….

}

## RBAC模型

RBAC模型（Role-Based Access Control：基于角色的访问控制）

就是用户通过角色与权限进行关联

简单的说，一个用户拥有若干角色，每个角色拥有若干权限

这样，就构成了\*\*“用户-角色-权限”\*\*的授权模型

在这种模型中，用户与角色之间，角色与权限之间，一般都是多对多关系

## svg图片设置颜色可以在svg的元素上添加fill=”#fff”来设置颜色

## 如何实现免登录

当有token时,直接跳转到首页

## 环境变量配置

### 使用环境变量

process.env.NODE\_ENV

NODE\_ENV就是在env文件中配置的变量,使用process.env获取

## DotENV插件

让.env文件中的代码高亮

## .env文件的作用

1. 环境变量的配置文件，简单说，就是在不同的环境下取不同的变量值

2．

.env.development(开发环境下,环境变量的配置) -> npm run dev / yarn dev -> 自动读取.env.development的值

.env.production(生产环境下,环境变量的配置) -> npm run build:prod/yarn build:prod -> 自动读取.env. production值

.env.staging(测试环境下,环境变量的配置) -> npm run build:stage/yarn build:stage -> 自动读取.env. staging的值

## 在业务代码和非业务代码中，使用环境变量的区别

业务代码指的是 ***src*** 目录中我们自己编写的项目源代码

\* 业务代码中

\* 业务代码中只能访问到 `NODE\_ENV` `BASE\_URL` 和以 `VUE\_APP\_` 打头的环境变量，其它变量不可识别。

\* 其中 `BASE\_URL`：

\* 业务代码中只能读取 publicPath 的值。

\* 非业务代码中可以读取到环境变量配置文件中设置的值。

\* 非业务代码中

\* 可以访问到环境变量配置文件中定义的所有环境变量。

## 深度选择器

/deep/ less语法深度选择器

::v-deep sass/scss深度选择器

回显数据 // 编辑表单时，将数据显示在输入框中

# Vue项目性能优化方案

1、v-for 正确设置key值  
2、封装复用的模块（http请求）、组件（ui库）  
3、路由懒加载：component：() => import('./xxx.vue')  
4、productionSourceMap: false  
5、启用gzip压缩，打包体积更小  
6、keep-alive 缓存不活跃组件  
7、插件CDN方式引入，减小项目体积  
8、图片使用CDN地址，图片懒加载

# 放大镜模板

<template>

  <div class="spec-preview">

    <img src="../assets/xiaoxin4.png" />

    <div class="event" @mousemove="moveMask"></div>

    <div class="big">

      <img src="../assets/xiaoxin4.png" ref="bigImg" />

    </div>

    <div class="mask" ref="maskRef"></div>

  </div>

</template>

<script>

export default {

  name: 'Magnifier',

  methods: {

    moveMask(e) {

      // 获取对象

      const bigImg = this.$refs.bigImg

      const mask = this.$refs.maskRef

      // 获取鼠标位置

      let x = e.offsetX - mask.offsetWidth / 2

      let y = e.offsetY - mask.offsetHeight / 2

      // 设置约束区域

      if (x < 0) x = 0

      if (x > mask.offsetWidth) x = mask.offsetWidth

      if (y < 0) y = 0

      if (y > mask.offsetHeight) y = mask.offsetHeight

      // 遮罩层跟随移动

      mask.style.left = x + 'px'

      mask.style.top = y + 'px'

      // 大图片移动

      bigImg.style.left = -2 \* x + 'px'

      bigImg.style.top = -2 \* y + 'px'

    },

  },

}

</script>

<style lang='less' scoped>

.spec-preview {

  position: relative;

  width: 400px;

  height: 400px;

  border: 1px solid #ccc;

  img {

    width: 100%;

    height: 100%;

  }

  .event {

    position: absolute;

    top: 0;

    left: 0;

    z-index: 998;

    width: 100%;

    height: 100%;

  }

  .mask {

    position: absolute;

    left: 0;

    top: 0;

    display: none;

    width: 50%;

    height: 50%;

    background-color: rgba(0, 255, 0, 0.3);

  }

  .big {

    position: absolute;

    top: -1px;

    left: 100%;

    width: 100%;

    height: 100%;

    border: 1px solid #aaa;

    z-index: 998;

    display: none;

    background: white;

    overflow: hidden;

    img {

      width: 200%;

      max-width: 200%;

      height: 200%;

      position: absolute;

      left: 0;

      top: 0;

    }

  }

  .event:hover ~ .mask,

  .event:hover ~ .big {

    display: block;

  }

}

</style>

# 分页

<template>

  <div class="box">

    <select v-model="size">

      <option value="5">5条数据</option>

      <option value="10">10条数据</option>

      <option value="15">15条数据</option>

    </select>

    {{startAndEndNum.start}}--{{startAndEndNum.end}}

    <button @click="$emit('changePageNo',pageNo-1)" :disabled="pageNo===1">上一页</button>

    <button v-if="startAndEndNum.start > 1" @click="$emit('changePageNo',1)">1</button>

    <button v-if="startAndEndNum.start > 2">...</button>

    <button v-for="item,i in startAndEndNum.end" :class="{active:pageNo===item}" @click="$emit('changePageNo',item)" v-if="item>=startAndEndNum.start"

      :key="i">{{item}}</button>

    <button v-if="startAndEndNum.end < maxPageNum - 1">...</button>

    <button v-if="startAndEndNum.end < maxPageNum" @click="$emit('changePageNo',maxPageNum)">{{maxPageNum}}</button>

    <button @click="$emit('changePageNo',pageNo+1)" :disabled="pageNo===maxPageNum">下一页</button>

    <input type="text" @keyup.enter="gotoPage" class="goto">

  </div>

</template>

<script>

export default {

  name: '',

  props: ['pageNo', 'pageSize', 'total', 'continues'],

  data() {

    return {

      size: this.pageSize,

    }

  },

  methods: {

    gotoPage(e) {

      let page = e.target.value

      if (isNaN(page) || page < 1 || page > this.maxPageNum) return (e.target.value = '')

      this.$emit('changePageNo', parseInt(page))

      e.target.value = ''

    },

  },

  computed: {

    // 最大页数

    maxPageNum() {

      return Math.ceil(this.total / this.size)

    },

    // 获取连续分页按钮的起始值和结束值

    startAndEndNum() {

      let start = 0,

        end = 0

      // 连续分页数大于页面总数

      if (this.continues > this.maxPageNum) {

        start = 1

        end = this.continues

      }

      // 连续分页数小于页面总数

      if (this.continues < this.maxPageNum) {

        start = this.pageNo - parseInt(this.continues / 2)

        end = this.pageNo + parseInt(this.continues / 2)

        // start 小于1

        if (start < 1) {

          start = 1

          end = this.continues

        }

        // end 大于最大页数

        if (end > this.maxPageNum) {

          end = this.maxPageNum

          start = this.maxPageNum - this.continues + 1

        }

      }

      return {

        start,

        end,

      }

    },

  },

}

</script>

<style lang='less' scoped>

.active {

  background-color: skyblue !important;

}

.box {

  margin: 40px auto;

  text-align: center;

  button {

    margin-left: 10px;

    padding: 0 10px;

    min-width: 30px;

    height: 30px;

    text-align: center;

    line-height: 30px;

    border: none;

    background-color: #ddd;

    cursor: pointer;

    user-select: none;

  }

  .goto {

    margin-left: 10px;

    width: 30px;

    height: 30px;

    border: 1px solid #ddd;

    outline: none;

  }

}

</style>