1. 弹窗组件的改进
   1. 预期效果



图1-1 弹窗输入框和权限选择子内容时权效果图



图1-2 弹窗有默认值时效果图



图1-3 弹窗上传图片时效果图



图1-4 弹窗添加和修改目录时效果图（必填项效果）

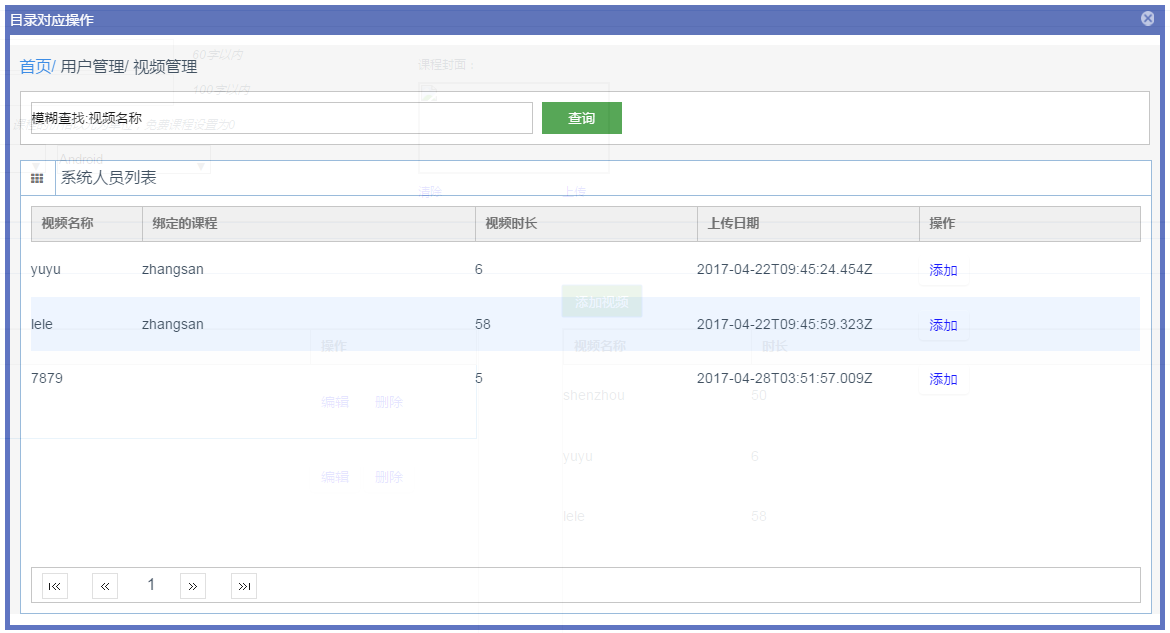


图1-5 弹窗添加对应目录视频时效果图

* 1. 掌握技能

表1-1 任务说明

|  |  |  |
| --- | --- | --- |
| 任务名称 | 子任务 | 任务内容 |
| 弹窗组件开发 | 1 | 开发一个适应多种场景的弹窗 |

* 1. 任务分解
     1. 在当前项目的/src/component路径下创建alert.vue文件
     2. 组件代码如下：

<template>

<div v-bind:style = "alertClass">

<div v-bind:style ="alertHol">

<div style=" height:30px;background:#4F66B3;margin-bottom:10px">

<span style="display:inline-block;float:left;color:white;line-height:30px;font-size:14px" v-text="AlertInputArr.alertName.value">管理员添加</span>

此处绑定弹窗的名字，比如“图片操作”“目录操作”等

<img v-on:click="AlertInputArr.btn\_click" style="display:inline-block;float:right;margin-top:5px;margin-right:5px" src="../assets/images/x.png">

</div>

<ul v-if="this.AlertInputArr.list.length">

这里的v-if是判断弹窗是否有表单项，如有时只有一句简单的提示“\*\*\*成功”

<template v-for="item in this.AlertInputArr.list">

这里的v-for是遍历给定的表单项数组的，比如一个有两个输入框和一个下拉框的弹窗数据，list中就包含三个数据

<li v-if ="item.type==='input'" style="display:block;margin-bottom:10px">

这里的v-if判断是否为输入框

<span style="display:inline-block;width:80px;text-align:right" v-html="item.rowName"></span>

这是输入框的前缀说明文字

<input\_default v-bind:width="item.width" height="20px" v-bind:defaultObj="item.defaultObj"></input\_default>

<span v-html="item.rowSupplement"></span>

这是后缀说明文字

</li>

<li v-if = "item.type==='powerSelector'"

这里的v-if判断是否为下拉框

style="display:block;height:30px;margin-bottom:10px;position:relative;top:0;left:0">

<span style="display:inline-block;width:80px;height:20px;line-height:20px;text-align:right;">权限选择:</span><div style="position:absolute;top:-5px;right:65px"><selector v-bind:selectorData="item"></selector></div>

</li>

<li v-if = "item.type==='img\_video\_Upload'||item.type==='img\_Upload'" v-bind:style="upload\_vedio\_li" style="position:relative">

这里的v-if判断是否为图片和视频上传

<span style="display:inline-block;width:80px;height:20px;line-height:20px;text-align:right;" v-html="item.rowName">视频上传:</span>

<template v-if = "uploadFlag">

这里片段是否为视频上传，因为图片上传和视频上传的处理逻辑是不一样的

<input\_default v-bind:width="item.width" height="20px" v-bind:defaultObj="item.defaultObj"></input\_default>

<a v-on:click="addPic($event,item)" v-text="item.aText" style="color:white;font-size:2px !important;text-decoration: none;width:50px;height:21px;line-height:22px;background-color:#4F66B3;float:right;margin-top:8px"></a>

</template>

<template v-else>

这里是图片上传

<img v-bind:src="item.imags.value" width="250" height="100" style="vertical-align:bottom;margin-bottom:5px" />

<a v-on:click="removeImage($event,item)" v-on:mouseenter = "img\_video\_btn" v-text="item.aTextC" style="color:white;font-size:2px !important;text-decoration: none;text-aligh:top;width:50px;height:21px;line-height:22px;background-color:#4F66B3;position:absolute;top:45px;right:-80px;margin-top:8px;border-radius:3px"></a>

<a v-on:click="uploadImage($event,item)" v-on:mouseenter = "img\_video\_btn" v-text="item.aTextU" style="color:white;font-size:2px !important;text-decoration: none;text-aligh:center;width:50px;height:21px;line-height:22px;background-color:#4F66B3;margin-top:8px;position:absolute;top:15px;right:-80px;border-radius:3px"></a>

</template>

<video v-if="item.type==='img\_video\_Upload'" v-bind:src = "item.videos.value" v-on:loadeddata="loadeddata($event,item)" v-show="false" id="videoId" controls="true">

</video>

这是一个隐藏的video标签，用来作为加载视频的标的

<input type="file" v-bind:accept="item.accept" v-bind:id="item.uuids" @change="onFileChange($event,item)" multiple style="display:none" >

这是一个隐藏的type为“file”的input，这是用来处理上传的表单控件，因样式太丑而被隐藏，由相应的a标签或button标签触发该input的change事件从而完成上传动作

</li>

</template>

</ul>

<div v-else>

<tableListVideo v-bind:isComponent="isComponentData"></tableListVideo>

</div>

<div style="text-align:center;" >

<btn v-for="key in AlertInputArr.btn" v-bind:props\_btn\_data\_obj="key"></btn>

</div>

</div>

</div>

</template>

<script type="text/javascript">

import input\_default from "./input\_default.vue";

import vedioListStore from "../vuex/vedioListStore.js";

import tableListVideo from "./addVideoList.vue";

import btn from "./btn.vue";

import selector from "./powerSelector.vue";

import store from "../vuex/store.js";

import axios from 'axios';

//允许跨域请求携带cookie

axios.defaults.withCredentials = true;

引入需要用到的组件

export default {

name: 'btn\_add',

data () {

return {

isComponentData:this.AlertDatas.isComponentData?this.AlertDatas.isComponentData:{iscomponent:true},

因为原本videoList.vue(视频页)是作为一个页面存在的，这里应用为组件的话需要处理一下它的头部和添加按钮不显示，该条数据就是这个作用

alertClass: {

'width':window.innerWidth+"px",

'height':window.innerHeight+"px",

background:"white",

'z-index':1050,

'position':'absolute',

'top':0,

'left':0,

'opacity':0.9

},

alertHol:{},

alertHolTop:{},

AlertInputArr:this.AlertDatas,

uploadFlag:true,

upload\_vedio\_li:{

'display':'block',

'height':'30px',

'margin-bottom':'10px',

'position':'relative'

}

}

},

props:['AlertDatas'],

弹窗组件具有较强的适应性，所以组件本身只提供一些基本的样式，而展示的数据和执行的动作都又外部传入，这个属性就是父组件向子组件传输具的。

computed:{

},

watch:{

uploadFlag:function(val){//监控上传控件切换重新计算弹窗高度

上传控件有缩略图功能，展示缩略图时弹窗的宽高需重新计算

if(!val){

this.alertHol.height=parseInt(this.alertHol.height)+70+"px";

this.upload\_vedio\_li=parseInt(this.upload\_vedio\_li.height)+70+"px";

}else{

this.alertHol.height=parseInt(this.alertHol.height)-70+"px";

this.upload\_vedio\_li=parseInt(this.upload\_vedio\_li.height)-70+"px";

}

}

},

mounted:function(){

该钩子函数是在组建初始化后执行的，我们在这里计算好弹窗的宽高

console.log(this.AlertDatas.isComponentData?this.AlertDatas.isComponentData:{iscomponent:true});

var width = (function(arr){

这个方法是遍历弹窗内所有控件，取宽度最宽的值用来计算弹窗的宽

if(arr.length>1){

for(var i = 1;i<arr.length;i++){

if(arr[i-1].width>arr[i].width){

var temp = arr[i-1].width;

arr[i-1].width = arr[i].width;

arr[i].width = temp;

}

};

return arr[arr.length-1].width;

}else if(arr.length===1){

return arr[0].width;

}else{

return 0;

}

})(this.AlertInputArr.list)+160;

这里的160是弹窗不算内含控件时的宽

var height = (function(arr){

该方法遍历弹窗所有控件，每个input或selector默认高30px,再加上弹窗不含控件的其余高度来计算弹窗的具体高度

var height=0;

if(arr.length>=1){

for(var i = 1;i<arr.length;i++){

if(arr[i].type==="input"||arr[i].type==="powerSelector"){

height+=30;

}

};

//return height;

}

return height;

})(this.AlertInputArr.list)+60+30+(this.AlertInputArr.btn.length===0||this.AlertInputArr.list.length<=1?20:130);

if(this.AlertInputArr.isComponentData&&!this.AlertInputArr.isComponentData.iscomponent){

width=1150;

height=620;

}

如果弹窗为给对应目录添加视频，因内容比较复杂，换用这种简单粗暴的重置宽高的算法

this.alertHol = {

'padding-top':0,

'position':'relative',

'background':'#FFFFFF',

'border':'5px solid #4F66BB',

'border-top':'none',

'width':width+'px',

'height':height+"px",

'top':window.innerHeight/2-height/2+'px',

'left':window.innerWidth/2-width/2+'px'

}

这个对象决定了弹窗的宽高和其他基本样式

this.alertHolTop = {

'height':"30px",

}

},

methods:{

getWidth:function(item){

if(!item){

return item.width;

}else{

return 0;

}

},

img\_video\_btn:function(e){

e.target.style.cursor = "pointer";

},

行内样式无法书写伪类，换成绑定鼠标的mouseenter等事件触发时更改样式来实现伪类的效果

addPic:function(e,item){//'选择按钮的单击事件'

e.preventDefault();

alert(item.uuids);

document.getElementById(item.uuids).click();

return false;

},

onFileChange(e,item) {//上传按钮的filechange事件

var files = e.target.files || e.dataTransfer.files;

if(item.type==="img\_video\_Upload"){

if (!files.length||files[0].size>1024\*1024\*1024)return;//如果未选择任何视频或视频大小超过1G就终断操作

//开始制作缩略图

var url = URL.createObjectURL(files[0]);

//改变video标签的src使其重新加载数据

item.videos.value=url;

// vedioListStore.state.AlertInputArr.list[0].defaultObj.defaultStr.value=files[0].name;

this.createImage(files,item);

}else{

if (!files.length||files[0].size>14\*1024\*1024)return;//如果未选择任何图片或图片大小超过4M就终断操作

//开始制作缩略图

var url = URL.createObjectURL(files[0]);

item.imags.value = url;

this.createImage(files,item);

}

},

createImage(files,item) {

if(typeof FileReader==='undefined'){

alert('您的浏览器不支持图片视频上传，请升级您的浏览器');

return false;

};

//控制选择框的显示与否

this.uploadFlag=false;

},

loadeddata:function(e,item){//视频标签加载完数据后触发的事件

if(item.type==="img\_video\_Upload"){

var video = e.target;

var canvas = document.createElement("canvas");

canvas.width = video.videoWidth\*0.3;

canvas.height = video.videoHeight\*0.3;

canvas.getContext('2d').drawImage(video, 0, 0, canvas.width, canvas.height);

item.imags.value=canvas.toDataURL("image/jpg");

}

},

removeImage: function(e,item) {

this.vedios = [];

this.uploadFlag=true;

//item.defaultObj.defaultStr.value="";

},

uploadImage: function(e,item) {

var \_this = this;

if(item.rowName.indexOf("视频上传")!==-1){

//利用表单对象进行上传

var form = new FormData();

if(document.getElementById(item.uuids).files[0]){

//参数为上传控件的id和该控件的数据

form.append(item.uuids,document.getElementById(item.uuids).files[0]);

}

axios.post(store.state.url.BASEURL+ store.state.url.VIDEOUPLOAD,form,{

method: 'post',

headers:{'Content-Type':'multipart/form-data'}

}).then(function (response) {

item.defaultObj.defaultStr.value=response.data.cacheName;

//恢复‘选择按钮’以及上传输入框

\_this.uploadFlag = true;

},function(response){

//恢复‘选择按钮’以及上传输入框

\_this.uploadFlag = true;

alert(response.data.err);

});

}else if(item.rowName.indexOf("图片上传")!==-1){

//判断item.\_this.a.state.uploadUrl.value是否为空，如不为空则清除该图片，因为不管是编辑还是重复上传，数据库中的上张图片已无用

if(item.\_this.a.state.uploadUrl.value){

var imgDropUrlArr = item.\_this.a.state.uploadUrl.value.split("=");

axios({

url:store.state.url.BASEURL+ store.state.url.DELETEDROPIMG+"&pathName="+imgDropUrlArr[imgDropUrlArr.length-1],

method: 'get'

}).then(function (response) {

console.log(response.data);

},function(response){

//恢复‘选择按钮’以及上传输入框

//\_this.uploadFlag = true;

// alert(response.data.err);

// item.\_this.a.state.uploadUrl.value="";

});

};

//利用表单对象进行上传

var form = new FormData();

if(document.getElementById(item.uuids).files[0]){

var render = function(src){

var img = new Image();

//img.id = item.uuids;

img.onload = function(e){

var canvas = document.createElement("canvas");

canvas.width = img.naturalWidth\*0.1;

canvas.height = img.naturalHeight\*0.1;

canvas.getContext('2d').drawImage(this, 0, 0, canvas.width, canvas.height);

//页面显示缩略图

item.imags.value=canvas.toDataURL("image/jpg");

var str = canvas.toDataURL("image/jpg");

str = str.substring(str.indexOf(",")+1,str.length);

var form = new FormData();

form.append("myId",document.getElementById(item.uuids).files[0]);

axios.post(store.state.url.BASEURL+ store.state.url.IMGUPLOAD,form,{

method: 'post',

headers:{'Content-Type':'multipart/form-data'}

}).then(function (response) {

console.log(response.data);

item.\_this.a.state.uploadUrl.value=store.state.url.BASEURL+response.data.cacheName;

item.\_this.a.state.uploadFlag.display="none";

console.log(item.\_this.a.state.uploadUrl.value);

},function(response){

//恢复‘选择按钮’以及上传输入框

\_this.uploadFlag = true;

// alert(response.data.err);

item.\_this.a.state.uploadUrl.value="";

});

//console.log(e.path[0].currentSrc);

};

img.src = src;

};

// 创建 FileReader 对象 并调用 render 函数来完成渲染.

var reader = new FileReader();

// 绑定load事件自动回调函数

reader.onload = function(e){

// 调用前面的 render 函数

render(e.target.result);

};

// 读取文件内容

reader.readAsDataURL(document.getElementById(item.uuids).files[0]);

}

}

}

},

beforeDestroy:function(){

},

components: {input\_default,btn,selector,tableListVideo}

}

</script>

<style>

.alert56HolId {

min-width:210px;

min-height:100px;

position:absolute;

top:0;

left:0;

background:#4F66B3;

opacity:0.8;

}

</style>

该组件大量使用了行内样式，因为行内样式优先级高，受vue影响小，在有大量交互和数据变化的组件内建议多用行内样式

* 1. 参考资料

无

* 1. 扩展练习

教师自行补充