iView + 七牛云 上传组件封装

作者：Parker
原文链接：https://ld246.com/article/1588684858157
来源网站：链滴

新建一个页面模板

```
<!--
 搞什么事的： 七牛云组件的封装，接口获取token(地址什么的都是在后端写好返回在token里了)
 用法：vue组件
 params:
   acceptType(0,对应data里fileTypes数组的下标)
   :refid="'myfile'"  (一个页面使用多个组件，用ref标示来区分)
   @next (向父组件传回调的next对象获取的参数)
   @complete (上传完成后返回对象(hash, key(七牛云路径))到父组件)
   @error (方法错误回调返回对象)

-->
<template>
  <div>
    <div class="buttonDiv">
      <Button
       icon="ios-cloud-upload-outline"
       @click="zh_uploadFile"
      >选择文件</Button>
      <input type="file" :ref="refid" @change="zh_uploadFile_change" style="display:none">
    </div>
<!--    进度条-->
    <div :ref="refid + 'ivu-progress'" class="ivu-progress ivu-progress-normal ivu-progress-sh
w-info" style="display: none;max-width: 350px;">
      <div class="ivu-progress-outer">
        <div class="ivu-progress-inner">
          <div class="ivu-progress-bg" :ref="refid + 'ivu-progress-bg'" style="height: 2px"></di
>
          <div class="ivu-progress-success-bg" :ref="refid + 'ivu-progress-success-bg'" style="w
dth: 0%; height: 2px;"></div>
        </div>
      </div>
      <span class="ivu-progress-text">
        <span class="ivu-progress-text-inner" :ref="refid + 'ivu-progress-text-inner'"></span>
      </span>
    </div>
  </div>
</template>

<style lang="less">
  .buttonDiv {
    display: inline-block;
  }
  .progress {
    width: 400px;
    height: 2px;
  }
</style>

<script>
    import * as qiniu from 'qiniu-js'
    import getQnyToken from "../../assets/script/qiniuyun/getQnyToken";
```

```
export default {
    name : 'qiNiuYun',
    props : {
        styles: Object,//样式
        refid:{//绑定的id
            type : String,
            default : ''
        },
        uploadType:{//上传类型 0：要素表    1：个人身份文件上传   2：证据文件上传
            type : String,
            default : ''
        },
        acceptType: {
            default: 0
        },
        path: ''  // 上传到七牛云到路径
    },
    data(){
        return {
            totalSize:0,//总量
            loaded:0,//已上传文件大小
            totalTime:0,//总用时/s
            speed:0,//当前速度
            lestTime:0,//剩余时间
            timmer:'',//定时器
            fileTypes: [
                ['.jpg', '.png', '.jpeg', '.txt', '.xls', '.xlsx', '.doc', '.docx', '.word', '.pdf', '.ppt', '.pptx', '.w
rdx'],  // 对应上传文件类型 acceptType = 0
            ]
        }
    },
    methods: {
        //选择上传文件
        zh_uploadFile(){
            this.$refs[this.refid].click();
        },

        //选择文件后触发的事件
        zh_uploadFile_change(evfile){
            let typeFlag = false
            const that=this
            const file = evfile.target.files[0] //Blob 对象，上传的文件
            if (file) {
                const fileTypes = this.fileTypes[this.acceptType]
                const fileend = file.name.substring(file.name.lastIndexOf('.'))
                for (let t of fileTypes) {
                    if (t === fileend) {
                        typeFlag = true
                        break
                    }
                }
            } else {
                this.$Message.error('取消上传！')
                return false
```

```
            }
            if (typeFlag) {
                //后端获取七牛token（//这里替换成给自己后端的token获取接口）
                getQnyToken().then(res=>{
                    const uptoken = res
                    this.totalSize=file.size;
                    const key = this.path + this.getuuid()+'-'+file.name // 上传后文件资源名以设置的key 为主，如果 key 为 null 或者 undefined，则文件资源名会以 hash 值作为资源名。
                    let config = {
                        useCdnDomain: true,  //表示是否使用 cdn 加速域名，为布尔值，true 表示使，默认为 false。
                        region: qiniu.region.z2,     // 根据具体提示修改上传地区,当为 null 或 undefined时，自动分析上传域名区域
                        concurrentRequestLimit: 7, //分片上传的并发请求量，number，默认为3
                    };

                    let putExtra = {
                        fname: "",  //文件原文件名
                        params: {}, //用来放置自定义变量
                        mimeType: null  //用来限制上传文件类型，为 null 时表示不对文件类型限制；制类型放到数组里：  ["image/png", "image/jpeg", "image/gif"]
                    };
                    this.observable = qiniu.upload(file, key, uptoken, putExtra, config)
                    //定时器用来统计上传时间
                    this.timmer=setInterval(() => {
                        //计时
                        that.totalTime+=1
                        //每秒统计一次速度（已上传大小/总耗时）
                        that.speed=that.loaded/that.totalTime
                        //计算剩余时间（总大小-已上传大小）/当前速度
                        that.lestTime=(that.totalSize-that.loaded)/that.speed
                    }, 1000);
                    this.subscription=this.observable.subscribe({
                        next: (result) => {
                            that.loaded=result.total.loaded
                            that.$emit('next',result,parseInt(this.lestTime)) // 可用来做进度条处理
                            // 百分比
                            let percent = result.total.percent.toFixed(0)
                            this.$refs[this.refid + 'ivu-progress'].style.display = 'block'
                            this.$refs[this.refid + 'ivu-progress-bg'].style.width = percent + '%'
                            this.$refs[this.refid + 'ivu-progress-text-inner'].innerHTML = percent + '%'
                            if (percent == '100') {
                                this.$refs[this.refid + 'ivu-progress'].style.display = 'none'
                            }
                            // 主要用来展示进度
                            // this.$Notice.info({
                            //     title: "上传中，请稍后....",
                            //     duration: 0,
                            //     render: h => {
                            //         return h('span', [
                            //             h('div', {
                            //                 style: {
                            //                     color: "red",
                            //                 },
```

```
                        //             }, `上传过程请勿关闭浏览器或执行其他操作！`),
                        //             h('div', `进度${this.flieProgress}%,预计剩余${this.leastTime}`),
                        //         ])
                        //     }
                        // })
                    },
                    error: (errResult) => {
                        that.$emit('error',errResult)
                    },
                    complete: (result) => {
                        window.clearInterval(this.timmer)//清除计时
                        result.fileName = file.name
                        that.$emit('complete',result)
                    }
                })
                //清除文本值方便下次触发change
                // this.$refs[this.refid].value = null
            })
        } else {
            this.$Message.error('指定文件格式错误，请重新上传')
        }
    },
    cancelUpload(){//取消上传
        window.clearInterval(this.timmer)//清除计时
        this.subscription.unsubscribe() // 上传取消
    },
    getuuid() {
        let s = []
        let hexDigits = "0123456789abcdef"
        for (let i = 0; i < 36; i++) {
            s[i] = hexDigits.substr(Math.floor(Math.random() * 0x10), 1)
        }
        s[14] = "4"  // bits 12-15 of the time_hi_and_version field to 0010
        s[19] = hexDigits.substr((s[19] & 0x3) | 0x8, 1);  // bits 6-7 of the clock_seq_hi_and_r
served to 01
        s[8] = s[13] = s[18] = s[23] = "-"

        let uuid = s.join("")
        return uuid
    },
    quotePercent(num, total) {
        num = parseFloat(num);
        total = parseFloat(total)
        if (isNaN(num) || isNaN(total)) {
            return '-'
        }
        return total <= 0 ? '0%' : (Math.round(num / total * 100) / 100.00)*100 + '%'
    },
    delFile(ref) {
        debugger
        this.$refs[ref].value = null
    }
  }
}
```

```
</script>
```

创建一个getQnyToken文件，来获取七牛云的上传token

```
import Axios from 'axios'

let api = process.env.baseUrl
let member_token
export default async function () {
  let store = $nuxt.$store
  // let route = $nuxt.$route
  if (process.browser) {
    member_token = JSON.parse(localStorage.getItem('memberToken'))
    // if (store.state.)
    let _token
    await Axios.get(api + '/api/qiniu/token/file?member_token=' + member_token)
      .then(res => {
        if (res.data.data) {
          _token = res.data.data.token
        }
      })
    return _token
  }
}
```

其他组件内调用：

```
<qi-niu-yun
            :acceptType=0
            :refid="'releaseTaskId'"
            :path="qnySaveUrl"
            v-show="qnyCallBackParam.documentName === null"
            ref="upload"
            @next="fileNext(arguments)"
            @complete="fileComplete"
            @error="fileError"/>
```

传入参数acceptType的值对应子组件（也就是封装的七牛云上传组件）里面filleTypes数组里面的下标

Props：

参数	说明	备注	类型
acceptType	对应子组件（也就是封装的七牛云上传组件）里面filleTypes数组里面的下标		int
refid	对应子组件的ref，唯一性	如果一个页面有个上传按钮，那么每个refid的值必须唯一	String
path	七牛云的上传路径		tring

Callback Events：

事件名	说明	返回类型
@next bj	向父组件传回调的next对象获取的参数	
@complete bj	上传完成后返回对象(hash, key(七牛云路径))到父组件	
@error bj	方法错误回调返回对象	