官网前端代码

<template>

<div class="wscn-http404-container">

<div class="wscn-http404">

<div class="pic-404">

<img class="pic-404\_\_parent" src="@/assets/404\_images/404.png" alt="404">

<img class="pic-404\_\_child left" src="@/assets/404\_images/404\_cloud.png" alt="404">

<img class="pic-404\_\_child mid" src="@/assets/404\_images/404\_cloud.png" alt="404">

<img class="pic-404\_\_child right" src="@/assets/404\_images/404\_cloud.png" alt="404">

</div>

<div class="bullshit">

<div class="bullshit\_\_oops">OOPS!</div>

<div class="bullshit\_\_info">All rights reserved

<a style="color:#20a0ff" href="https://wallstreetcn.com" target="\_blank">wallstreetcn</a>

</div>

<div class="bullshit\_\_headline">{{ message }}</div>

<div class="bullshit\_\_info">Please check that the URL you entered is correct, or click the button below to return to the homepage.</div>

<a href="" class="bullshit\_\_return-home">Back to home</a>

</div>

</div>

</div>

</template>

<script>

export default {

name: 'Page404',

computed: {

message() {

return 'The webmaster said that you can not enter this page...'

}

}

}

</script>

<style lang="scss" scoped>

.wscn-http404-container{

transform: translate(-50%,-50%);

position: absolute;

top: 40%;

left: 50%;

}

.wscn-http404 {

position: relative;

width: 1200px;

padding: 0 50px;

overflow: hidden;

.pic-404 {

position: relative;

float: left;

width: 600px;

overflow: hidden;

&\_\_parent {

width: 100%;

}

&\_\_child {

position: absolute;

&.left {

width: 80px;

top: 17px;

left: 220px;

opacity: 0;

animation-name: cloudLeft;

animation-duration: 2s;

animation-timing-function: linear;

animation-fill-mode: forwards;

animation-delay: 1s;

}

&.mid {

width: 46px;

top: 10px;

left: 420px;

opacity: 0;

animation-name: cloudMid;

animation-duration: 2s;

animation-timing-function: linear;

animation-fill-mode: forwards;

animation-delay: 1.2s;

}

&.right {

width: 62px;

top: 100px;

left: 500px;

opacity: 0;

animation-name: cloudRight;

animation-duration: 2s;

animation-timing-function: linear;

animation-fill-mode: forwards;

animation-delay: 1s;

}

@keyframes cloudLeft {

0% {

top: 17px;

left: 220px;

opacity: 0;

}

20% {

top: 33px;

left: 188px;

opacity: 1;

}

80% {

top: 81px;

left: 92px;

opacity: 1;

}

100% {

top: 97px;

left: 60px;

opacity: 0;

}

}

@keyframes cloudMid {

0% {

top: 10px;

left: 420px;

opacity: 0;

}

20% {

top: 40px;

left: 360px;

opacity: 1;

}

70% {

top: 130px;

left: 180px;

opacity: 1;

}

100% {

top: 160px;

left: 120px;

opacity: 0;

}

}

@keyframes cloudRight {

0% {

top: 100px;

left: 500px;

opacity: 0;

}

20% {

top: 120px;

left: 460px;

opacity: 1;

}

80% {

top: 180px;

left: 340px;

opacity: 1;

}

100% {

top: 200px;

left: 300px;

opacity: 0;

}

}

}

}

.bullshit {

position: relative;

float: left;

width: 300px;

padding: 30px 0;

overflow: hidden;

&\_\_oops {

font-size: 32px;

font-weight: bold;

line-height: 40px;

color: #1482f0;

opacity: 0;

margin-bottom: 20px;

animation-name: slideUp;

animation-duration: 0.5s;

animation-fill-mode: forwards;

}

&\_\_headline {

font-size: 20px;

line-height: 24px;

color: #222;

font-weight: bold;

opacity: 0;

margin-bottom: 10px;

animation-name: slideUp;

animation-duration: 0.5s;

animation-delay: 0.1s;

animation-fill-mode: forwards;

}

&\_\_info {

font-size: 13px;

line-height: 21px;

color: grey;

opacity: 0;

margin-bottom: 30px;

animation-name: slideUp;

animation-duration: 0.5s;

animation-delay: 0.2s;

animation-fill-mode: forwards;

}

&\_\_return-home {

display: block;

float: left;

width: 110px;

height: 36px;

background: #1482f0;

border-radius: 100px;

text-align: center;

color: #ffffff;

opacity: 0;

font-size: 14px;

line-height: 36px;

cursor: pointer;

animation-name: slideUp;

animation-duration: 0.5s;

animation-delay: 0.3s;

animation-fill-mode: forwards;

}

@keyframes slideUp {

0% {

transform: translateY(60px);

opacity: 0;

}

100% {

transform: translateY(0);

opacity: 1;

}

}

}

}

</style>

<template>

<div id="app">

<router-view />

</div>

</template>

<script>

export default {

name: 'App',

updated() {

window.scroll(0, 0);

},

}

</script>

<template>

<section class="app-main">

<transition name="fade-transform" mode="out-in">

<router-view :key="key" />

</transition>

</section>

</template>

<script>

export default {

name: "AppMain",

computed: {

key() {

return this.$route.path;

}

}

};

</script>

<style scoped lang="scss">

.app-main {

min-height: calc(100vh - 50px);

width: 100%;

position: relative;

overflow: hidden;

background-color: rgba(255, 244, 243, 0.22);

}

.fixed-header + .app-main {

padding-top: 50px;

}

.el-popup-parent--hidden {

.fixed-header {

padding-right: 15px;

}

}

</style>

<template>

<div>

<div v-if="!isMobile">

<div class="banner">

<img src="../../assets/article\_fill.jpg" alt="" />

</div>

<el-row :gutter="20">

<el-col :span="10" style="margin-left: 360px" class="main\_container">

<h1 class="main\_title">{{ temp.videoTitle }}</h1>

<div class="article\_info">

<span>{{ temp.createAt }}</span>

<span style="padding-left: 8px"

>&emsp; 作者：{{ temp.videoAuthor }}</span

>

</div>

<div class="video\_detail">

<iframe

:src="videoProfileTemp"

scrolling="no"

border="0"

frameborder="no"

framespacing="0"

width="800px"

height="500px"

allowfullscreen="true"

sandbox="allow-top-navigation allow-same-origin allow-forms allow-scripts"

>

</iframe>

</div>

<div class="video\_info">视频简介：{{ temp.videoProfile }}</div>

</el-col>

<el-col :span="4" :offset="1">

<div class="block-title">

推荐视频

</div>

<el-row

v-for="item in list"

class="abbr\_article\_single"

:key="item.id"

>

<router-link :to="'/video/' + item.id">

<div class="abbr\_article\_word">

<div class="abbr\_article\_title">

{{ item.videoTitle }}

</div>

<div class="abbr\_article\_info">

<span>{{ item.createAt }}</span>

<span style="padding-left: 8px"

>&emsp; 作者：{{ item.videoAuthor }}</span

>

</div>

</div>

<img alt="" :src="item.videoPic" class="abbr\_article\_img" />

</router-link>

</el-row>

</el-col>

</el-row>

</div>

<div v-else>

<div>

<img src="../../assets/article\_fill.jpg" alt="" class="full\_a" />

</div>

<div class="card">

<el-card

shadow="never"

style="margin-bottom: 16px;padding: 0px;"

:body-style="{ padding: '10px' }"

>

<h1 class="main\_title">{{ temp.videoTitle }}</h1>

<div class="article\_info">

<span>{{ temp.createAt }}</span>

<span style="padding-left: 8px"

>&emsp; 作者：{{ temp.videoAuthor }}</span

>

</div>

<div style="text-align: center;margin: 38px 0 16px; ">

<div style="position: relative; padding: 28.25% 45%;">

<iframe

:src="videoProfileTemp"

scrolling="no"

border="0"

frameborder="no"

framespacing="0"

allowfullscreen="true"

sandbox="allow-top-navigation allow-same-origin allow-forms allow-scripts"

style="position: absolute; width: 100%; height: 100%; left: 0; top: 0;"

>

</iframe>

</div>

</div>

<div class="video\_info">视频简介：{{ temp.videoProfile }}</div>

</el-card>

<el-card

shadow="never"

style="margin-bottom: 16px;"

class="t"

:body-style="{ padding: '10px' }"

>

<div class="t\_title">

推荐视频

</div>

<el-row

v-for="item in list"

style="width: 100%;margin-top: 16px;"

:key="item.id"

:gutter="24"

>

<router-link :to="'/video/' + item.id">

<el-col :span="17" style="padding-left: 18px;">

<div style="display: inline-block;">

<div class="video\_title\_c">

{{ item.videoTitle }}

</div>

<div class="abbr\_article\_info">

<span>{{ item.createAt }}</span>

<br />

<span>作者：{{ item.videoAuthor }}</span

>

</div>

</div>

</el-col>

<el-col :span="7" style="padding-left: 24px;">

<img alt="" :src="item.videoPic" class="t\_img" />

</el-col>

</router-link>

</el-row>

</el-card>

</div>

</div>

</div>

</template>

<script>

import WebApi from "@/api/WebApi";

import { mapState } from "vuex";

export default {

name: "detail",

data() {

return {

total: 0,

videoProfileTemp: "",

temp: {

readings: "",

videoAuthor: "",

videoUrl: ``,

videoTitle: "",

videoProfile: "",

status: "",

createAt: "",

isDel: ""

},

listTemp: ,

list: null,

listQuery: {

page: 1,

size: 5,

keywords: "",

direction: "DESC"

}

};

},

created() {

const id = this.$route.params && this.$route.params.id;

this.fetchData(id);

this.getRefresh();

},

methods: {

fetchData(id) {

WebApi.findOneVideo(id)

.then(data => {

this.temp = data;

console.log(this.temp);

})

.catch(err => {

console.log(err);

});

},

getRefresh() {

WebApi.findAllVideo(this.listQuery).then(data => {

this.list = data.data;

this.total = this.list.length;

Object.keys(this.list).forEach(

key =>

(this.list.videoPic =

process.env.VUE\_APP\_API + "/file/" + this.list.videoPic)

);

this.videoProfileTemp =

"https://player.bilibili.com/player.html?aid=" +

this.temp.videoUrl +

"&page=1&high\_quality=1&danmaku=0";

});

}

},

computed: {

...mapState("app", )

}

};

</script>

<style lang="scss">

@media screen and (max-width: 768px) {

.video\_info {

font-size: 16px;

color: #656565;

margin-bottom: 0 !important;

}

.card {

padding: 16px;

}

.t {

padding: 10px;

.t\_title {

font-size: 18px;

color: #222;

border-bottom: 1px solid #e5e9ef;

padding-bottom: 10px;

}

.t\_img {

display: inline-block;

height: 74px;

border-radius: 4px;

overflow: hidden;

vertical-align: middle;

background-size: cover;

background-position: 50%;

background-repeat: no-repeat;

}

}

.abbr\_article\_word {

width: 100%;

}

.abbr\_article\_single {

width: 100%;

}

.video\_title\_c {

font-size: 12px;

color: #222;

line-height: 1.5;

max-height: 32px;

overflow: hidden;

}

.abbr\_article\_info{

margin-top: 8px !important;

margin-left: 2px;

line-height: 1.5;

}

.full\_a {

width: 100%;

padding: 0.5rem 0.5rem 0 0.5rem;

}

}

.main\_container {

margin-top: 40px;

}

.main\_title {

width: 100%;

/\*height: 30px;\*/

overflow: hidden;

text-overflow: ellipsis;

font-size: 22px;

font-weight: bold;

line-height: 30px;

color: #323232;

/\*text-align: center;\*/

}

.banner {

display: block;

margin: 60px auto 0;

border-radius: 0.6rem;

width: 1200px;

height: 150px;

overflow: hidden;

}

.article\_info {

float: left;

/\*margin-bottom: 20px;\*/

width: 100%;

height: 14px;

overflow: hidden;

text-overflow: ellipsis;

font-size: 14px;

line-height: 14px;

color: #a2a2a2;

/\*text-align: center;\*/

}

.video\_detail {

display: block;

/\*margin: 15px 10px 0 10px;\*/

padding-top: 50px;

margin-bottom: 30px;

}

.block-title {

display: block;

margin-top: 70px;

font-size: 18px;

color: #222;

border-bottom: 1px solid #e5e9ef;

padding-bottom: 10px;

}

.abbr\_article\_single {

margin-top: 16px;

width: 310px;

display: block;

font-size: 0;

}

.abbr\_article\_word {

width: 250px;

line-height: 16px;

display: inline-block;

font-size: 12px;

color: #222;

margin-right: 12px;

vertical-align: middle;

}

.abbr\_article\_title {

max-height: 32px;

transition: 0.3s;

text-overflow: ellipsis;

overflow: hidden;

display: -webkit-box;

-webkit-box-orient: vertical;

}

.abbr\_article\_info {

font-size: 12px;

color: #99a2aa;

letter-spacing: 0;

margin-top: 4px;

}

.abbr\_article\_img {

display: inline-block;

/\*width: 68px;\*/

height: 50px;

border-radius: 4px;

overflow: hidden;

vertical-align: middle;

background-size: cover;

background-position: 50%;

background-repeat: no-repeat;

}

.video\_info {

font-size: 16px;

/\*color: #99a2aa;\*/

letter-spacing: 0;

margin-bottom: 80px;

margin-left: 5px;

line-height: 25px;

}

</style>

<template>

<div class="article\_detail">

<div v-if="!isMobile">

<div class="banner">

<img src="../../assets/article\_fill.jpg" alt="" />

</div>

<el-row :gutter="20">

<el-col :span="9" :offset="5" class="main\_container">

<h1 class="main\_title\_c">{{ temp.title }}</h1>

<div class="article\_info">

<span>{{ temp.createAt }}</span>

<span style="padding-left: 8px"

>&emsp; 作者：{{ temp.author }}</span

>

</div>

<div class="article\_detail" v-html="temp.article"></div>

</el-col>

<el-col :span="4" style="margin-left: 30px">

<div class="block-title">

推荐文章

</div>

<el-row

v-for="item in list"

class="abbr\_article\_single"

:key="item.id"

>

<router-link :to="'/article/' + item.id">

<div class="abbr\_article\_word">

<div class="abbr\_article\_title">

{{ item.title }}

</div>

<div class="abbr\_article\_info">

<span>{{ item.createAt }}</span>

<span style="padding-left: 8px"

>&emsp; 作者：{{ item.author }}</span

>

</div>

</div>

<img alt="" :src="item.pic" class="abbr\_article\_img" />

</router-link>

</el-row>

</el-col>

</el-row>

</div>

<div v-else>

<div>

<img src="../../assets/article\_fill.jpg" alt="" class="full" />

</div>

<div class="card">

<el-card shadow="never" style="margin-bottom: 16px;padding: 10px;">

<h1 class="main\_title\_c">{{ temp.title }}</h1>

<div class="article\_info">

<span>{{ temp.createAt }}</span>

<span style="padding-left: 8px"

>&emsp; 作者：{{ temp.author }}</span

>

</div>

<div class="article\_detail " v-html="temp.article"></div>

</el-card>

<el-card shadow="never" style="margin-bottom: 16px;" class="t">

<div class="t\_title">

推荐文章

</div>

<el-row

v-for="item in list"

class="abbr\_article\_single"

style="width: 100%"

:key="item.id"

:gutter="24"

>

<router-link :to="'/article/' + item.id">

<el-col :span="18">

<div class="abbr\_article\_word">

<div class="abbr\_article\_title">

{{ item.title }}

</div>

<div class="abbr\_article\_info">

<span>{{ item.createAt }}</span>

<span style="padding-left: 8px"

>&emsp; 作者：{{ item.author }}</span

>

</div>

</div>

</el-col>

<el-col :span="6">

<img alt="" :src="item.pic" class="abbr\_article\_img" />

</el-col>

</router-link>

</el-row>

</el-card>

</div>

</div>

</div>

</template>

<script>

import WebApi from "@/api/WebApi";

import { mapState } from "vuex";

export default {

name: "detail",

data() {

return {

total: 0,

temp: {

readings: "",

author: "",

article: ``,

title: "",

status: "",

createAt: "",

isDel: ""

},

listTemp: ,

list: null,

listQuery: {

page: 1,

size: 5,

keywords: "",

direction: "DESC"

}

};

},

created() {

const id = this.$route.params && this.$route.params.id;

this.fetchData(id);

this.getRefresh();

},

methods: {

fetchData(id) {

WebApi.findOneArticle(id)

.then(data => {

this.temp = data;

})

.catch(err => {

console.log(err);

});

},

getRefresh() {

WebApi.findSomeArticle(this.listQuery).then(data => {

this.list = data.data;

this.total = this.list.length;

Object.keys(this.list).forEach(

key =>

(this.list.pic =

process.env.VUE\_APP\_API + "/file/" + this.list.pic)

);

Object.keys(this.list).forEach(

key => (this.list.article = this.list.article + "……")

);

});

}

},

computed: {

...mapState("app", )

}

};

</script>

<style lang="scss">

.article\_detail {

@media screen and (max-width: 768px) {

.card {

padding: 10px;

.article,

.article\_contain {

padding: 0 10px 1px;

}

.card\_body {

margin-bottom: 16px;

}

}

.abbr\_article\_word{

width: 5.5rem !important;

}

.article\_detail {

padding-bottom: 0 !important;

section {

img {

width: auto;

height: auto;

max-width: 100%;

max-height: 100%;

}

}

p {

img {

width: auto;

height: auto;

max-width: 100%;

max-height: 100%;

}

}

}

.t {

padding: 10px;

.t\_title {

font-size: 18px;

color: #222;

border-bottom: 1px solid #e5e9ef;

padding-bottom: 10px;

}

}

.full {

width: 100%;

padding: 0.5rem 0.5rem 0 0.5rem;

}

.abbr\_article\_img{

/\*margin-left: 16px;\*/

}

}

.main\_container {

margin-top: 60px;

}

.main\_title\_c {

width: 100%;

font-size: 22px;

font-weight: bold;

line-height: 30px;

color: #323232;

text-align: center;

}

.banner {

display: block;

margin: 60px auto 0;

border-radius: 0.6rem;

width: 1200px;

height: 150px;

overflow: hidden;

}

.article\_info {

float: left;

/\*margin-bottom: 20px;\*/

width: 100%;

height: 14px;

overflow: hidden;

text-overflow: ellipsis;

font-size: 14px;

line-height: 14px;

color: #a2a2a2;

text-align: center;

}

.article\_detail {

display: block;

float: left;

/\*margin: 15px 10px 0 10px;\*/

width: 100%;

font-size: 16px;

line-height: 28px;

color: #656565;

padding-bottom: 50px;

}

.block-title {

display: block;

margin-top: 70px;

font-size: 18px;

color: #222;

border-bottom: 1px solid #e5e9ef;

padding-bottom: 10px;

}

.abbr\_article\_single {

margin-top: 16px;

width: 310px;

display: block;

font-size: 0;

}

.abbr\_article\_word {

width: 230px;

line-height: 16px;

display: inline-block;

font-size: 12px;

color: #222;

margin-right: 12px;

vertical-align: middle;

}

.abbr\_article\_title {

max-height: 32px;

transition: 0.3s;

text-overflow: ellipsis;

overflow: hidden;

display: -webkit-box;

-webkit-box-orient: vertical;

}

.abbr\_article\_info {

font-size: 12px;

color: #99a2aa;

letter-spacing: 0;

margin-top: 4px;

}

.abbr\_article\_img {

display: inline-block;

/\*width: 68px;\*/

height: 50px;

border-radius: 4px;

// margin-left: 16px;

overflow: hidden;

vertical-align: middle;

background-size: cover;

background-position: 50%;

background-repeat: no-repeat;

}

}

</style>

<template>

<el-footer style="height:120px;" class="footer">

<div class="cpr">

<div style="padding-top: 30px;">

<span>Copyright@ 2017-2020 广东米果传媒有限公司·版权所有</span>

</div>

<a class="icp" href="http://beian.miit.gov.cn" target="\_blank">

<span>粤ICP备20019636号-1</span>

</a>

</div>

</el-footer>

</template>

<script>

export default {

name: "EndPart"

}

</script>

<style lang="scss">

.cpr{

display: block;

text-align: center;

color: #8a8a8a;

line-height: 30px;

}

.el-footer {

background-color: #373737;

color: #333;

text-align: center;

line-height: 60px;

}

.footer{

@media screen and (max-width: 768px){

font-size: 12px;

}

@media screen and (min-width: 768px){

font-size: 16px;

}

}

</style>

<template>

<div v-if="!item.hidden" class="menu-wrapper">

<template v-if="hasOneShowingChild(item.children,item) && (!onlyOneChild.children||onlyOneChild.noShowingChildren)&&!item.alwaysShow">

<app-link v-if="onlyOneChild.meta" :to="resolvePath(onlyOneChild.path)">

<el-menu-item :index="resolvePath(onlyOneChild.path)" class="submenu-title">

<item :icon="onlyOneChild.meta.icon||(item.meta&&item.meta.icon)" :title="onlyOneChild.meta.title" />

</el-menu-item>

</app-link>

</template>

<el-submenu v-else ref="subMenu" :index="resolvePath(item.path)" popper-append-to-body>

<template slot="title">

<item v-if="item.meta" :icon="item.meta && item.meta.icon" :title="item.meta.title" />

</template>

<head-nav

v-for="child in item.children"

:key="child.path"

:is-nest="true"

:item="child"

:base-path="resolvePath(child.path)"

class="nest-menu"

/>

</el-submenu>

</div>

</template>

<script>

import path from 'path'

import { isExternal } from '@/utils/validate'

import Item from './Item'

import AppLink from './Link'

export default {

name: 'HeadNav',

components: { Item, AppLink },

props: {

// route object

item: {

type: Object,

required: true

},

isNest: {

type: Boolean,

default: false

},

basePath: {

type: String,

default: ''

}

},

data() {

// To fix https://github.com/PanJiaChen/vue-admin-template/issues/237

// TODO: refactor with render function

this.onlyOneChild = null

return {}

},

methods: {

hasOneShowingChild(children = , parent) {

const showingChildren = children.filter(item => {

if (item.hidden) {

return false

} else {

// Temp set(will be used if only has one showing child)

this.onlyOneChild = item

return true

}

})

// When there is only one child router, the child router is displayed by default

if (showingChildren.length === 1) {

return true

}

// Show parent if there are no child router to display

if (showingChildren.length === 0) {

this.onlyOneChild = { ... parent, path: '', noShowingChildren: true }

return true

}

return false

},

resolvePath(routePath) {

if (isExternal(routePath)) {

return routePath

}

if (isExternal(this.basePath)) {

return this.basePath

}

return path.resolve(this.basePath, routePath)

}

}

}

</script>

<template>

<div class="hello">

<h1>{{ msg }}</h1>

<p>

For a guide and recipes on how to configure / customize this project,<br>

check out the

<a href="https://cli.vuejs.org" target="\_blank" rel="noopener">vue-cli documentation</a>.

</p>

<h3>Installed CLI Plugins</h3>

<ul>

<li><a href="https://github.com/vuejs/vue-cli/tree/dev/packages/%40vue/cli-plugin-babel" target="\_blank" rel="noopener">babel</a></li>

<li><a href="https://github.com/vuejs/vue-cli/tree/dev/packages/%40vue/cli-plugin-eslint" target="\_blank" rel="noopener">eslint</a></li>

</ul>

<h3>Essential Links</h3>

<ul>

<li><a href="https://vuejs.org" target="\_blank" rel="noopener">Core Docs</a></li>

<li><a href="https://forum.vuejs.org" target="\_blank" rel="noopener">Forum</a></li>

<li><a href="https://chat.vuejs.org" target="\_blank" rel="noopener">Community Chat</a></li>

<li><a href="https://twitter.com/vuejs" target="\_blank" rel="noopener">Twitter</a></li>

<li><a href="https://news.vuejs.org" target="\_blank" rel="noopener">News</a></li>

</ul>

<h3>Ecosystem</h3>

<ul>

<li><a href="https://router.vuejs.org" target="\_blank" rel="noopener">vue-router</a></li>

<li><a href="https://vuex.vuejs.org" target="\_blank" rel="noopener">vuex</a></li>

<li><a href="https://github.com/vuejs/vue-devtools#vue-devtools" target="\_blank" rel="noopener">vue-devtools</a></li>

<li><a href="https://vue-loader.vuejs.org" target="\_blank" rel="noopener">vue-loader</a></li>

<li><a href="https://github.com/vuejs/awesome-vue" target="\_blank" rel="noopener">awesome-vue</a></li>

</ul>

</div>

</template>

<script>

export default {

name: 'HelloWorld',

props: {

msg: String

}

}

</script>

<!-- Add "scoped" attribute to limit CSS to this component only -->

<style scoped>

h3 {

margin: 40px 0 0;

}

ul {

list-style-type: none;

padding: 0;

}

li {

display: inline-block;

margin: 0 10px;

}

a {

color: #42b983;

}

</style>

<template>

<div class="app-wrapper">

<sidebar class="sidebar-container" />

<app-main />

<end-part />

</div>

</template>

<script>

import { AppMain,Sidebar,EndPart } from './components'

export default {

name: "Layout",

components: {

Sidebar,

AppMain,

EndPart

},

}

</script>

<style lang="scss" scoped>

@import "~@/styles/mixin.scss";

@import "~@/styles/variables.scss";

.drawer-bg {

background: #000;

opacity: 0.3;

width: 100%;

top: 0;

height: 100%;

position: absolute;

z-index: 999;

}

.fixed-header {

position: fixed;

top: 0;

right: 0;

z-index: 9;

width: calc(100% - #{$sideBarWidth});

transition: width 0.28s;

}

.hideSidebar .fixed-header {

width: calc(100% - 54px)

}

.mobile .fixed-header {

width: 100%;

}

</style>

<template>

<div class="article\_list">

<div v-if="!isMobile">

<swiper-vertical />

<div class="filter-container">

<el-input

class="filter-item"

placeholder="请输入内容"

prefix-icon="el-icon-search"

v-model="listQuery.keywords"

>

</el-input>

<el-button

class="filter-item-button"

type="primary"

icon="el-icon-search"

@click="getRefresh"

>

搜索

</el-button>

</div>

<el-row type="flex" justify="center">

<el-col v-if="list" :span="12" class="article\_container">

<el-row v-for="item in list" class="article\_single" :key="item.id">

<router-link :to="'/article/' + item.id">

<div class="article\_word">

<div class="article\_title">

{{ item.title }}

</div>

<div class="article\_info">

<span>{{ item.createAt }}</span>

<span style="padding-left: 8px"

>&emsp; 作者：{{ item.author }}</span

>

</div>

<div class="article\_profile" v-html="item.article"></div>

</div>

<img alt="" :src="item.pic" class="article\_img" />

</router-link>

</el-row>

</el-col>

</el-row>

<pagination

v-show="total || total > 0"

:total="total"

:page.sync="listQuery.page"

:size.sync="listQuery.size"

@pagination="getRefresh"

/>

</div>

<div v-else>

<my-swiper />

<div class="search">

<el-input

class="search\_input"

placeholder="请输入内容"

prefix-icon="el-icon-search"

v-model="listQuery.keywords"

>

</el-input>

<el-button

class="search\_btn"

type="primary"

icon="el-icon-search"

@click="getRefresh"

>

搜索

</el-button>

</div>

<div class="card">

<el-card

shadow="never"

v-for="(item, idx) in list"

:key="idx"

:body-style="{ padding: '0px' }"

class="card\_body"

@click.native="$router.push(`/article/${item.id}`)"

>

<img :src="item.pic" class="full" />

<div class="article">

<h5 class="article\_title">

{{ item.title }}

</h5>

</div>

<div class="article\_info">

<span> {{ item.createAt }}</span>

<span style="padding-left: 8px"

>&emsp; 作者：{{ item.author }}</span

>

</div>

<div class="article\_contain" v-html="item.article"></div>

</el-card>

</div>

<pagination

v-show="total || total > 0"

:total="total"

:page.sync="listQuery.page"

:size.sync="listQuery.size"

@pagination="getRefresh"

class="pagination"

layout="total,prev, pager, next,jumper"

/>

</div>

</div>

</template>

<script>

import SwiperVertical from "@/components/SwiperVertical";

import MySwiper from "@/components/Swiper";

import WebApi from "@/api/WebApi";

import Pagination from "@/components/Pagination"; // secondary package based on el-pagination

import { mapState } from "vuex";

export default {

name: "Article",

components: { SwiperVertical, Pagination, MySwiper },

data() {

return {

total: 0,

index: "",

createAt: "",

listTemp: ,

list: null,

listQuery: {

page: 1,

size: 8,

keywords: "",

direction: "DESC"

}

};

},

created() {

this.getRefresh();

},

methods: {

getRefresh() {

WebApi.findSomeArticle(this.listQuery).then(data => {

this.list = data.data;

this.total = this.list.length;

Object.keys(this.list).forEach(

key =>

(this.list.pic =

process.env.VUE\_APP\_API + "/file/" + this.list.pic)

);

Object.keys(this.list).forEach(

key => (this.list.article = this.list.article + "……")

);

});

},

handleClick() {

console.log("成立");

}

},

computed: {

...mapState("app", )

}

};

</script>

<style lang="scss">

.article\_list {

@media screen and (min-width: 768px) {

.article\_single {

background: rgba(241, 255, 248, 0.6);

/\*float: left;\*/

display: block;

margin-bottom: 30px;

border-radius: 0.6rem;

/\*z-index: 3;\*/

box-shadow: 0 0.06rem 0.12rem rgba(0, 0, 0, 0.1),

0 0.2rem 0.6rem rgba(0, 0, 0, 0.1);

/\*transition-duration: 0.3s;\*/

overflow: hidden;

}

.article\_title {

padding-top: 25px;

padding-left: 30px;

width: 658px;

overflow: hidden;

font-size: 16px;

color: #323232;

}

.article\_word {

float: left;

width: 658px;

/\*height: 120px;\*/

overflow: hidden;

margin-right: 30px;

}

.article\_info {

clear: both;

display: block;

margin-top: 15px;

margin-left: 30px;

line-height: 12px;

font-size: 12px;

color: #a2a2a2;

}

.article\_profile {

float: left;

/\*margin-top: 5px;\*/

padding-left: 30px;

width: 658px;

overflow: hidden;

font-size: 14px;

color: #656565;

line-height: 24px;

}

.article\_img {

float: left;

width: 212px;

overflow: hidden;

}

.article\_container {

width: 900px;

}

}

@media screen and (max-width: 768px) {

.el-pagination {

white-space: inherit !important;

}

.card {

padding: 16px;

.article,

.article\_contain {

padding: 0 10px 0;

}

.article\_contain {

font-size: 12px;

line-height: 20px;

color: #656565;

}

.card\_body {

margin-bottom: 16px;

}

.article\_title {

margin-top: 0.6rem;

margin-bottom: 0.3rem;

width: 100%;

line-height: 24px;

font-size: 18px;

color: #313131;

overflow: hidden;

}

.article\_info {

font-size: 12px;

color: #a2a2a2;

/\*text-align: center;\*/

margin-left: 0.3rem;

}

}

.search {

display: flex;

justify-content: space-around;

margin-top: 16px;

box-sizing: border-box;

padding: 0 12px;

.search\_input {

width: 60%;

}

.search\_btn {

width: 30%;

}

}

.full {

width: 100%;

}

}

}

</style>

<template>

<div>

<el-row type="flex" justify="center">

<el-col :md="8" :sm="24">

<div class="intro">

<h3>关于我们</h3>

<div class="con2 contact2">

<h4>公司简介</h4>

<div class="op">我们是腾讯在肇庆市的城市合作伙伴，提供企业和政府品牌建设服务、企业和政府宣传片摄制、大型活动策划与组织。在公司发展壮大的3年里，我们始终为客户提供好的产品和技术支持、健全的售后服务，如果您对我公司的产品服务有兴趣，期待您在线留言或者来电咨询。</div>

</div>

<div class="con contact">

<h4>公司宣言</h4>

<p>让世界看到您的价值！</p>

</div>

<div class="con contact">

<h4>大众平台</h4>

<el-row class="img\_com" type="flex" justify="center">

<div>

<img src="../../assets/qrcode/sina.jpg" alt="">

<p>新浪微博</p>

</div>

<div style="padding-left: 10px">

<img src="../../assets/qrcode/wechat.jpg" alt="">

<p>公众号</p>

</div>

<div style="padding-left: 10px">

<img src="../../assets/qrcode/mini.jpg" alt="">

<p>小程序</p>

</div>

<div style="padding-left: 10px">

<img src="../../assets/qrcode/toutiao.jpg" alt="">

<p>今日头条</p>

</div>

</el-row>

<el-row class="img\_com" type="flex" justify="center">

<div>

<img src="../../assets/qrcode/bilibili.jpg" alt="">

<p>哔哩哔哩</p>

</div>

<div style="padding-left: 10px">

<img src="../../assets/qrcode/douyin.jpg" alt="">

<p>抖音</p>

</div>

</el-row>

</div>

<h3>联系方式</h3>

<div class="con contact">

<h4>广东米果传媒有限公司</h4>

<p>肇庆市端州区肇庆学院中巴软件园综合办公实验楼505室<br>

电话：13827527862&#x3000;邮编：526000</p>

</div>

</div>

</el-col>

</el-row>

</div>

</template>

<script>

export default {

name: "index"

}

</script>

<style scoped lang="scss">

.intro{

margin-top: 50px;

}

.intro h3 {

font-size: 30px;

font-weight: bold;

text-align: center;

background: url(../../assets/t-line.png) center 7px no-repeat;

margin-bottom: 30px;

}

.intro h4 {

font-size: 18px;

color: #323232;

text-align: center;

margin: 25px 0 20px;

}

.intro .con p {

text-align: center;

font-size: 14px;

color: #656565;

line-height: 25px;

/\*text-indent: 2em;\*/

}

.intro .con2 .op {

padding-left: 50px;

padding-right: 50px;

font-size: 14px;

color: #656565;

line-height: 25px;

/\*text-indent: 2em;\*/

}

.img\_com {

/\*width: 60%;\*/

/\*margin: 0 20%;\*/

// overflow: hidden;

margin-top: 30px;

flex-wrap: wrap;

}

.img\_com img{

width:150px;

text-align: center;

}

</style>

<template>

<div>

<my-swiper />

<recommend />

<grid-switch />

<irregular-container />

<!-- <div class="dashboard-container"></div>-->

</div>

</template>

<script>

import MySwiper from "@/components/Swiper";

import GridSwitch from "@/components/GridSwitch";

import IrregularContainer from "@/components/IrregularContainer";

import Recommend from '@/components/Recommend'

export default {

components: { MySwiper, GridSwitch,IrregularContainer,Recommend },

data () {

return {

name: '123'

}

},

name: 'Home',

computed: {

}

}

</script>

<style lang="scss" scoped>

.dashboard {

&-container {

margin: 80px;

}

&-text {

font-size: 30px;

line-height: 46px;

}

}

</style>

<script>

export default {

created() {

const { params, query } = this.$route

const { path } = params

this.$router.replace({ path: '/' + path, query })

},

render: function(h) {

return h() // avoid warning message

}

}

</script>

<template>

<div class="video\_list">

<div v-if="!isMobile">

<swiper-vertical />

<div class="filter-container">

<el-input

class="filter-item"

placeholder="请输入内容"

prefix-icon="el-icon-search"

v-model="listQuery.keywords"

>

</el-input>

<el-button

class="filter-item-button"

type="primary"

icon="el-icon-search"

@click="getRefresh"

>

搜索

</el-button>

</div>

<el-row

:gutter="20"

v-for="(row, i) in listTemp"

:key="i"

type="flex"

class="row-bg"

justify="center"

style="padding-bottom:50px;"

>

<el-col

:span="3"

:id="'T\_' + (i \* 3 + j)"

v-for="(cell, j) in row"

:key="cell.id"

:offset="j > 0 ? 1 : 0"

>

<router-link :to="'/video/' + cell.id">

<el-card :body-style="{ padding: '0px' }">

<img :src="cell.videoPic" class="image" alt="" />

<div class="video\_title\_a">

<span>{{ cell.videoTitle }}</span>

<div class="bottom clearfix">

<!-- <span class="info">{{ cell.videoProfile }}</span>-->

</div>

</div>

</el-card>

</router-link>

</el-col>

</el-row>

<pagination

v-show="total || total > 0"

:total="total"

:page.sync="listQuery.page"

:size.sync="listQuery.size"

@pagination="getRefresh"

/>

</div>

<div v-else>

<my-swiper />

<div class="search">

<el-input

class="search\_input"

placeholder="请输入内容"

prefix-icon="el-icon-search"

v-model="listQuery.keywords"

>

</el-input>

<el-button

class="search\_btn"

type="primary"

icon="el-icon-search"

@click="getRefresh"

>

搜索

</el-button>

</div>

<el-row :gutter="24" style="margin-top: 16px;padding: 16px; margin-bottom: -50px;">

<el-col

v-for="(item, idx) in list"

:key="idx"

:span="12"

style="min-height: 9rem;"

>

<el-card :body-style="{ padding: '0px' }" @click.native="$router.push(`/video/${item.id}`)">

<img :src="item.videoPic" style="width: 100%;" alt="" />

<div class="video\_title\_a">

<span>{{ item.videoTitle }}</span>

</div>

</el-card>

</el-col>

</el-row>

<pagination

v-show="total || total > 0"

:total="total"

:page.sync="listQuery.page"

:size.sync="listQuery.size"

@pagination="getRefresh"

layout="total,prev, pager, next,jumper"

/>

</div>

</div>

</template>

<script>

import SwiperVertical from "@/components/SwiperVertical";

import WebApi from "@/api/WebApi";

import MySwiper from "@/components/Swiper";

import Pagination from "@/components/Pagination"; // secondary package based on el-pagination

import { mapState } from "vuex";

export default {

name: "Video",

components: { MySwiper, SwiperVertical, Pagination },

data() {

return {

total: 0,

index: "",

createAt: "",

listTemp: ,

list: null,

listQuery: {

page: 1,

size: 8,

keywords: "",

direction: "DESC"

}

};

},

created() {

this.getRefresh();

},

methods: {

getRefresh() {

WebApi.findAllVideo(this.listQuery).then(data => {

this.list = data.data;

this.total = this.list.length;

Object.keys(this.list).forEach(

key =>

(this.list.videoPic =

process.env.VUE\_APP\_API + "/file/" + this.list.videoPic)

);

// console.log(this.list)

const \_list = this.list,

listTemp = ,

sectionCount = 4;

for (const i in \_list) {

const index = parseInt((i / sectionCount).toString());

if (listTemp.length <= index) {

listTemp.push();

}

listTemp.push(\_list);

}

this.listTemp = listTemp;

});

}

},

computed: {

...mapState("app", )

}

};

</script>

<style lang="scss">

.video\_list {

@media screen and (max-width: 768px) {

.el-pagination {

white-space: inherit !important;

}

.card {

padding: 16px;

.article,

.article\_contain {

padding: 0 10px 0;

}

.article\_contain {

font-size: 12px;

line-height: 20px;

color: #656565;

}

.card\_body {

margin-bottom: 16px;

}

.article\_title {

margin-top: 20px;

margin-bottom: 10px;

width: 100%;

line-height: 24px;

font-size: 18px;

color: #313131;

overflow: hidden;

}

.article\_info {

font-size: 12px;

color: #a2a2a2;

text-align: center;

}

}

.search {

display: flex;

justify-content: space-around;

margin-top: 16px;

box-sizing: border-box;

padding: 0 12px;

.search\_input {

width: 60%;

}

.search\_btn {

width: 30%;

}

}

.full {

width: 100%;

}

.video\_title\_a {

padding: 12px;

font-size: 14px;

line-height: 1.2;

}

}

@media screen and (min-width: 768px) {

.image {

width: 100%;

display: block;

}

.info {

font-size: 13px;

color: #999;

}

.bottom {

margin-top: 13px;

line-height: 12px;

}

.video\_title\_a {

padding: 16px;

font-size: 16px;

line-height: 23px;

}

}

}

</style>

<template>

<div>

<el-row v-if="!isMobile" type="flex" justify="center" style="margin-bottom: 80px;">

<el-col v-if="listTemp" :span="7" class="articleL">

<el-row class="articleL1">

<router-link :to="'/article/' + listTemp.id">

<img alt="" :src="listTemp.pic" class="article1\_img" />

<div class="article1">

<h5 class="article\_title">

{{ listTemp.title }}

</h5>

<span class="article\_contain\_i" v-html="listTemp.article">

</span>

</div>

</router-link>

</el-row>

<el-row class="articleL2">

<router-link :to="'/article/' + listTemp.id">

<div class="article2">

<h5 class="article\_title">

{{ listTemp.title }}

</h5>

<span class="article\_contain\_i" v-html="listTemp.article">

</span>

</div>

<img alt="" :src="listTemp.pic" class="article2\_img" />

</router-link>

</el-row>

<el-row class="articleL3">

<router-link :to="'/article/' + listTemp.id">

<img alt="" :src="listTemp.pic" class="article3\_img" />

<div class="article3">

<h5 class="article\_title">

{{ listTemp.title }}

</h5>

<span class="article\_contain\_i" v-html="listTemp.article">

</span>

</div>

</router-link>

</el-row>

</el-col>

<el-col v-if="listTemp" :span="7" class="articleC">

<el-row class="articleC1">

<router-link :to="'/article/' + listTemp.id">

<img alt="" :src="listTemp.pic" class="article4\_img" />

<div class="article4">

<h5 class="article\_title">

{{ listTemp.title }}

</h5>

<span class="article\_contain\_j" v-html="listTemp.article">

</span>

</div>

</router-link>

</el-row>

<el-row class="articleC2">

<router-link :to="'/article/' + listTemp.id">

<div class="article5">

<h5 class="article\_title">

{{ listTemp.title }}

</h5>

<span class="article\_contain\_j" v-html="listTemp.article">

</span>

</div>

<img alt="" :src="listTemp.pic" class="article5\_img" />

</router-link>

</el-row>

</el-col>

<el-col v-if="listTemp" :span="7" class="articleR">

<el-row class="articleR1">

<router-link :to="'/article/' + listTemp.id">

<img alt="" :src="listTemp.pic" class="article6\_img" />

<div class="article6">

<h5 class="article\_title">

{{ listTemp.title }}

</h5>

<span class="article\_contain" v-html="listTemp.article">

</span>

</div>

</router-link>

</el-row>

<el-row class="articleR2">

<router-link :to="'/article/' + listTemp.id">

<img alt="" :src="listTemp.pic" class="article7\_img" />

<div class="article7">

<h5 class="article\_title">

{{ listTemp.title }}

</h5>

<span class="article\_contain" v-html="listTemp.article">

</span>

</div>

</router-link>

</el-row>

</el-col>

</el-row>

<div v-else class="card">

<el-card

shadow="never"

v-for="(item, idx) in list"

:key="idx"

:body-style="{ padding: '0px' }"

class="card\_body"

@click.native="$router.push(`/article/${item.id}`)"

>

<img :src="item.pic" class="full" />

<div class="article">

<h5 class="article\_title">

{{ item.title }}

</h5>

</div>

<span class="article\_contain" v-html="item.article"></span>

</el-card>

</div>

</div>

</template>

<script>

import WebApi from "@/api/WebApi";

import { mapState } from "vuex";

export default {

name: "IrregularContainer",

data() {

return {

total: "",

listTemp: null,

list: ,

listQuery: {

page: 1,

size: 7,

keywords: "",

direction: "DESC"

}

};

},

created() {

this.getRefresh();

},

methods: {

getRefresh() {

WebApi.findSomeArticle(this.listQuery).then(data => {

this.list = data.data;

this.total = this.list.length;

Object.keys(this.list).forEach(

key =>

(this.list.pic =

process.env.VUE\_APP\_API + "/file/" + this.list.pic)

);

Object.keys(this.list).forEach(

key => (this.list.article = this.list.article + "……")

);

// console.log(this.list)

const \_list = this.list,

listTemp = ;

for (const i in \_list) {

const index = parseInt((i / 3).toString());

if (listTemp.length <= index) {

listTemp.push();

}

listTemp.push(\_list);

}

this.listTemp = listTemp;

// console.log(this.listTemp)

});

}

},

computed: {

...mapState("app", )

}

};

</script>

<style lang="scss" scoped>

.articleL {

float: left;

margin-right: 20px;

width: 570px;

}

.articleL1,

.articleL3,

.articleL2 {

/\*height: 240px;\*/

box-shadow: 0 0 14px #e4e2e2;

-webkit-box-shadow: 0 0 14px #e4e2e2;

}

.article1\_img,

.article3\_img {

display: block;

float: left;

margin-right: 20px;

max-height: 240px;

}

.article1,

.article3 {

display: block;

float: left;

width: 230px;

padding-right: 20px;

overflow: hidden;

}

.article\_title {

margin-top: 20px;

margin-bottom: 2px;

width: 100%;

/\*height: 18px;\*/

line-height: 24px;

font-size: 18px;

color: #313131;

overflow: hidden;

}

.article\_contain {

margin-top: 5px;

display: block;

float: left;

width: 100%;

font-size: 12px;

line-height: 20px;

color: #656565;

overflow: hidden;

}

.article2\_img {

display: block;

float: right;

margin-left: 20px;

max-height: 240px;

}

.article2 {

display: block;

float: left;

width: 230px;

padding-left: 20px;

overflow: hidden;

}

.articleL1,

.articleC1,

.articleC2,

.articleR1 {

margin-top: 20px;

display: block;

float: left;

width: 100%;

background: #fff;

overflow: hidden;

}

.articleR2 {

margin-top: 30px;

display: block;

float: left;

width: 100%;

background: #fff;

overflow: hidden;

}

.articleL2,

.articleL3 {

margin-top: 44px;

display: block;

float: left;

width: 100%;

background: #fff;

overflow: hidden;

}

.articleC {

float: left;

margin-right: 20px;

width: 260px;

}

.articleC1 {

width: 100%;

/\*height: 370px;\*/

box-shadow: 0 0 14px #e4e2e2;

-webkit-box-shadow: 0 0 14px #e4e2e2;

}

.articleC2 {

width: 100%;

height: 370px;

box-shadow: 0 0 14px #e4e2e2;

-webkit-box-shadow: 0 0 14px #e4e2e2;

}

.articleR2 {

width: 100%;

/\*height: 360px;\*/

box-shadow: 0 0 14px #e4e2e2;

-webkit-box-shadow: 0 0 14px #e4e2e2;

}

.article4\_img {

display: block;

float: left;

margin-right: 20px;

max-width: 260px;

}

.article5\_img {

margin: 10px 0 0;

width: 260px;

/\*position: absolute;\*/

left: 0;

bottom: 0;

}

.article6\_img {

display: block;

float: left;

width: 320px;

}

.article7\_img {

display: block;

float: left;

width: 320px;

margin-top: -30px;

}

.article4,

.article5 {

display: block;

float: left;

/\*width: 220px;\*/

padding: 0 20px;

overflow: hidden;

}

.article6,

.article7 {

display: block;

float: left;

/\*width: 300px;\*/

padding: 0 20px;

overflow: hidden;

}

.articleR {

float: left;

margin-right: 20px;

width: 320px;

}

.articleR1 {

width: 100%;

/\*height: 380px;\*/

box-shadow: 0 0 14px #e4e2e2;

-webkit-box-shadow: 0 0 14px #e4e2e2;

}

.article\_contain\_i {

display: block;

float: left;

width: 100%;

font-size: 14px;

line-height: 22px;

color: #656565;

overflow: hidden;

}

.article\_contain\_j {

display: block;

float: left;

width: 100%;

font-size: 14px;

line-height: 20px;

color: #656565;

overflow: hidden;

}

@media screen and (max-width: 768px) {

.card {

padding: 16px;

.article, .article\_contain {

padding: 0 10px 0;

margin-top: 0 !important;

}

.article\_contain{

font-size: 15px;

}

.card\_body {

margin-bottom: 16px;

}

}

.full {

width: 100%;

}

}

</style>

<template>

<div :class="{'hidden':hidden}" class="pagination-container">

<el-pagination

:background="background"

:current-page.sync="currentPage"

:page-size.sync="pageSize"

:layout="layout"

:page-sizes="pageSizes"

:total="total"

v-bind="$attrs"

@size-change="handleSizeChange"

@current-change="handleCurrentChange"

/>

</div>

</template>

<script>

import { scrollTo } from '@/utils/scroll-to'

export default {

name: 'Pagination',

props: {

total: {

required: true,

type: Number

},

page: {

type: Number,

default: 1

},

limit: {

type: Number,

default: 8

},

pageSizes: {

type: Array,

default() {

return

}

},

layout: {

type: String,

default: 'total, sizes, prev, pager, next, jumper'

},

background: {

type: Boolean,

default: true

},

autoScroll: {

type: Boolean,

default: true

},

hidden: {

type: Boolean,

default: false

}

},

computed: {

currentPage: {

get() {

return this.page

},

set(val) {

this.$emit('update:page', val)

}

},

pageSize: {

get() {

return this.limit

},

set(val) {

this.$emit('update:limit', val)

}

}

},

methods: {

handleSizeChange(val) {

this.$emit('pagination', { page: this.currentPage, limit: val })

if (this.autoScroll) {

scrollTo(0, 800)

}

},

handleCurrentChange(val) {

this.$emit('pagination', { page: val, limit: this.pageSize })

if (this.autoScroll) {

scrollTo(0, 800)

}

}

}

}

</script>

<style scoped>

.pagination-container {

text-align: center;

/\*background: #fff;\*/

/\*padding: 32px 16px;\*/

padding-bottom: 50px;

}

.pagination-container.hidden {

display: none;

}

</style>

<template>

<el-row style="margin: 24px 24px;">

<el-col :sm="24" :md="{ span: '16', offset: '4' }">

<el-card>

<el-row :gutter="16">

<el-col :sm="24" :md="8" style="margin-bottom: 16px;">

<h2 class="title" style="line-height: 1.5;">

{{ video.videoTitle }}

</h2>

<div class="info">

<span style="margin-right: 16px;">{{ video.createAt }}</span>

<span>{{ video.videoAuthor }}</span>

</div>

<div class="introduction">

{{ video.videoProfile }}

</div>

</el-col>

<el-col :sm="24" :md="16">

<div style="position: relative; padding: 28.25% 45%;">

<iframe

:src="videoUrl"

scrolling="no"

border="0"

frameborder="no"

framespacing="0"

allowfullscreen="true"

sandbox="allow-top-navigation allow-same-origin allow-forms allow-scripts"

style="position: absolute; width: 100%; height: 100%; left: 0; top: 0;"

>

</iframe>

</div>

</el-col>

</el-row>

</el-card>

</el-col>

</el-row>

</template>

<script>

import WebApi from "@/api/WebApi";

import { mapState } from "vuex";

export default {

data() {

return {

video: {},

videoUrl: ""

};

},

mounted() {

this.fetch();

},

methods: {

async fetch() {

let data = await WebApi.findOneVideo("414807535779065856");

console.log(data);

this.video = data;

this.videoUrl =

"https://player.bilibili.com/player.html?aid=" +

data.videoUrl +

"&page=1&high\_quality=1&danmaku=0&as\_wide=1";

// console.log(this.videoUrl)

// this.videoUrl = 'https://player.bilibili.com/player.html?cid=145147963&aid=84267566&page=1&as\_wide=1&high\_quality=1&danmaku=0'

}

},

computed: {

...mapState("app", )

}

};

</script>

<style lang="scss" scoped>

.info {

// text-align: center;

font-size: 18px;

color: #656565;

}

.introduction {

font-size: 14px;

color: #aaa;

line-height: 1.5;

margin-top: 24px;

}

</style>

<template>

<div v-if="isExternal" :style="styleExternalIcon" class="svg-external-icon svg-icon" v-on="$listeners" />

<svg v-else :class="svgClass" aria-hidden="true" v-on="$listeners">

<use :href="iconName" />

</svg>

</template>

<script>

// doc: https://panjiachen.github.io/vue-element-admin-site/feature/component/svg-icon.html#usage

import { isExternal } from '@/utils/validate'

export default {

name: 'SvgIcon',

props: {

iconClass: {

type: String,

required: true

},

className: {

type: String,

default: ''

}

},

computed: {

isExternal() {

return isExternal(this.iconClass)

},

iconName() {

return `#icon-${this.iconClass}`

},

svgClass() {

if (this.className) {

return 'svg-icon ' + this.className

} else {

return 'svg-icon'

}

},

styleExternalIcon() {

return {

mask: `url(${this.iconClass}) no-repeat 50% 50%`,

'-webkit-mask': `url(${this.iconClass}) no-repeat 50% 50%`

}

}

}

}

</script>

<style scoped>

.svg-icon {

width: 1em;

height: 1em;

vertical-align: -0.15em;

fill: #337ab7;

overflow: hidden;

}

.svg-external-icon {

background-color: currentColor;

mask-size: cover!important;

display: inline-block;

}

</style>

<template>

<div>

<el-carousel

v-if="!isMobile"

:interval="5000"

arrow="always"

trigger="click"

height="450px"

>

<el-carousel-item v-for="item in list" :key="item.id">

<img :src="item.swiperPic" class="banner\_img" alt="" />

</el-carousel-item>

</el-carousel>

<swiper v-else class="home\_swiper" :options="swiperOption">

<swiper-slide class="swiper-item" v-for="banner in list" :key="banner.id">

<img :src="banner.swiperPic" class="full" />

</swiper-slide>

<div class="swiper-button-prev" slot="button-prev"></div>

<div class="swiper-button-next" slot="button-next"></div>

</swiper>

</div>

</template>

<script>

import WebApi from "@/api/WebApi";

import { mapState } from "vuex";

export default {

data() {

return {

newArr: ,

total: "",

swiperPic: "",

list: ,

swiperOption: {

spaceBetween: 30,

centeredSlides: true,

autoplay: {

delay: 2500,

disableOnInteraction: false

},

navigation: {

nextEl: ".swiper-button-next",

prevEl: ".swiper-button-prev"

}

}

};

},

created() {

this.getRefresh();

},

methods: {

getRefresh() {

WebApi.findAllSwiper().then(data => {

this.list = data;

Object.keys(this.list).forEach(

key =>

(this.list.swiperPic =

process.env.VUE\_APP\_API + "/file/" + this.list.swiperPic)

);

});

}

},

computed: {

...mapState('app',)

}

};

</script>

<style lang="scss" scoped>

.home\_swiper {

// max-height: 450px;

.swiper-item {

height: 100%;

text-align: center;

font-size: 18px;

background: #fff;

display: flex;

justify-content: center;

align-items: center;

}

}

.full {

width: 100%;

height: 100%;

}

</style>

<template>

<el-carousel :interval="5000" type="card" trigger="click" height="450px">

<el-carousel-item v-for="item in list" :key="item.id">

<img :src="item.swiperPic" class="banner\_img" alt=""/>

</el-carousel-item>

</el-carousel>

<!-- <el-carousel height="150px">-->

<!-- <el-carousel-item v-for="item in 4" :key="item">-->

<!-- <h3 class="small">{{ item }}</h3>-->

<!-- </el-carousel-item>-->

<!-- </el-carousel>-->

</template>

<script>

import WebApi from "@/api/WebApi";

export default {

name: "Swiper",

data() {

return {

newArr: ,

total: '',

swiperPic: '',

list:

}

},

created() {

this.getRefresh()

},

methods: {

getRefresh(){

WebApi.findAllSwiper().then(data => {

this.list = data

Object.keys(this.list).forEach(key => (this.list.swiperPic = process.env.VUE\_APP\_API + '/file/' +this.list.swiperPic))

})

}

}

}

</script>

<template>

<el-carousel :interval="5000" direction="vertical" arrow="always" trigger="click" height="450px">

<el-carousel-item v-for="item in list" :key="item.id">

<img :src="item.swiperPic" class="banner\_img" alt=""/>

</el-carousel-item>

</el-carousel>

<!-- <el-carousel height="150px">-->

<!-- <el-carousel-item v-for="item in 4" :key="item">-->

<!-- <h3 class="small">{{ item }}</h3>-->

<!-- </el-carousel-item>-->

<!-- </el-carousel>-->

</template>

<script>

import WebApi from "@/api/WebApi";

export default {

name: "Swiper",

data() {

return {

newArr: ,

total: '',

swiperPic: '',

list:

}

},

created() {

this.getRefresh()

},

methods: {

getRefresh(){

WebApi.findAllSwiper().then(data => {

this.list = data

Object.keys(this.list).forEach(key => (this.list.swiperPic = process.env.VUE\_APP\_API + '/file/' +this.list.swiperPic))

})

}

}

}

</script>

<template>

<div>

<el-menu

:default-active="activeIndex"

:background-color="variables.menuBg"

:text-color="variables.menuText"

:active-text-color="variables.menuActiveText"

mode="horizontal"

>

<img class="logo" src="@/assets/web\_logo.png" alt=""/>

<head-nav v-for="route in permission\_routes" :key="route.path" :item="route" :base-path="route.path" />

</el-menu>

</div>

</template>

<script>

import { mapGetters } from 'vuex'

import HeadNav from './HeadNav'

import variables from '@/styles/variables.scss'

export default {

components: { HeadNav },

computed: {

...mapGetters([

'permission\_routes',

]),

activeIndex() {

const route = this.$route

const {meta, path} = route

// if set path, the sidebar will highlight the path you set

if (meta.activeMenu) {

return meta.activeMenu

}

return path

},

variables() {

return variables

},

}

}

</script>

<template>

<div>

<el-carousel

v-if="!isMobile && listTemp"

:interval="5000"

type="card"

height="470px"

style="padding: 40px 0 10px 0;"

>

<el-carousel-item v-for="(row, i) in listTemp" :key="i">

<el-row type="flex" class="row-bg" justify="center">

<el-col

:span="7"

:id="'T\_' + (i \* 3 + j)"

v-for="(cell, j) in row"

:key="cell.id"

:offset="j > 0 ? 1 : 0"

>

<router-link :to="'/video/' + cell.id">

<el-card :body-style="{ padding: '0px' }">

<img :src="cell.videoPic" class="image" alt="" />

<div class="video\_title">

<span>{{ cell.videoTitle }}</span>

</div>

</el-card>

</router-link>

</el-col>

</el-row>

</el-carousel-item>

</el-carousel>

<swiper

class="swiper"

:options="swiperOption"

ref="swiper"

@click-slide="handleClick"

v-if="isMobile"

>

<swiper-slide v-for="(row, i) in list" :key="i">

<el-card :body-style="{ padding: '0px' }">

<img :src="row.videoPic" class="full" />

<div class="video\_title">

<span>{{ row.videoTitle }}</span>

</div>

</el-card>

</swiper-slide>

<div class="swiper-pagination " slot="pagination"></div>

<div class="swiper-button-prev" slot="button-prev"></div>

<div class="swiper-button-next" slot="button-next"></div>

</swiper>

</div>

</template>

<script>

import WebApi from "@/api/WebApi";

import { mapState } from "vuex";

export default {

name: "GridSwitch",

data() {

return {

total: "",

index: "",

videoPic: "",

videoTitle: "",

videoProfile: "",

listTemp: null,

list: ,

listQuery: {

page: 1,

size: 8,

keywords: "",

direction: "DESC"

},

swiperOption: {

effect: "coverflow",

grabCursor: true,

centeredSlides: true,

slidesPerView: "2",

coverflowEffect: {

rotate: 50,

stretch: 0,

depth: 100,

modifier: 1,

slideShadows: false

},

pagination: {

el: ".swiper-pagination"

},

autoplay: {

delay: 2500,

disableOnInteraction: false

},

navigation: {

nextEl: ".swiper-button-next",

prevEl: ".swiper-button-prev"

}

}

};

},

computed: {

// listTemp() {

// const \_list = this.list, listTemp = , sectionCount = 3;

// for (const i in \_list) {

// const index = parseInt((i / sectionCount).toString());

// if (listTemp.length <= index) {

// listTemp.push();

// }

// listTemp.push(\_list);

// }

// return listTemp;

// }

...mapState("app", )

},

created() {

this.getRefresh();

},

methods: {

getRefresh() {

WebApi.findAllVideo(this.listQuery).then(data => {

this.list = data.data;

this.total = this.list.length;

Object.keys(this.list).forEach(

key =>

(this.list.videoPic =

process.env.VUE\_APP\_API + "/file/" + this.list.videoPic)

);

// console.log(this.list)

const \_list = this.list,

listTemp = ,

sectionCount = 2;

for (const i in \_list) {

const index = parseInt((i / sectionCount).toString());

if (listTemp.length <= index) {

listTemp.push();

}

listTemp.push(\_list);

}

this.listTemp = listTemp;

});

},

handleClick(data) {

this.$router.push(`/video/${this.list.id}`);

}

}

};

</script>

<style lang="scss">

@media screen and (min-width: 768px) {

.image {

width: 100%;

display: block;

}

.info {

font-size: 13px;

color: #999;

}

.bottom {

margin-top: 13px;

line-height: 12px;

}

.el-carousel\_\_mask {

opacity: 0;

}

.video\_title {

padding: 10px 10px 5px;

font-size: 14px;

}

}

@media screen and (max-width: 768px) {

.swiper {

padding-bottom: 24px;

.swiper-item {

height: 100%;

text-align: center;

font-size: 16px;

background: #fff;

display: flex;

justify-content: center;

align-items: center;

}

}

.full {

display: block;

width: 100%;

}

.swiper-pagination {

bottom: 0px !important;

}

.video\_title {

padding: 10px 10px 5px;

font-size: 14px;

}

}

</style>

<script>

export default {

name: 'MenuItem',

functional: true,

props: {

icon: {

type: String,

default: ''

},

title: {

type: String,

default: ''

}

},

render(h, context) {

const { icon, title } = context.props

const vnodes =

if (icon) {

vnodes.push(<svg-icon icon-class={icon}/>)

}

if (title) {

vnodes.push(<span slot='title'>{(title)}</span>)

}

return vnodes

}

}

</script>

<template>

<!-- eslint-disable vue/require-component-is -->

<component v-bind="linkProps(to)">

<slot />

</component>

</template>

<script>

import { isExternal } from '@/utils/validate'

export default {

props: {

to: {

type: String,

required: true

}

},

methods: {

linkProps(url) {

if (isExternal(url)) {

return {

is: 'a',

href: url,

target: '\_blank',

rel: 'noopener'

}

}

return {

is: 'router-link',

to: url

}

}

}

}

</script>

const state = {

device: 'desktop',

isMobile: document.body.clientWidth <= 768? true: false

}

const mutations = {

TOGGLE\_DEVICE: (state, device) => {

state.device = device

}

}

const actions = {

toggleDevice({ commit }, device) {

commit('TOGGLE\_DEVICE', device)

}

}

export default {

namespaced: true,

state,

mutations,

actions

}

import Cookies from 'js-cookie'

const TokenKey = 'vue\_admin\_template\_token'

export function getToken() {

return Cookies.get(TokenKey)

}

export function setToken(token) {

return Cookies.set(TokenKey, token)

}

export function removeToken() {

return Cookies.remove(TokenKey)

}

/\*eslint-disable\*/

// 把 YYYY-MM-DD 改成了 yyyy-MM-dd

(function (main) {

'use strict';

/\*\*

\* Parse or format dates

\* @class fecha

\*/

var fecha = {};

var token = /d{1,4}|M{1,4}|yy(?:yy)?|S{1,3}|Do|ZZ|()\1?||"\*"|'\*'/g;

var twoDigits = /\d\d?/;

var threeDigits = /\d{3}/;

var fourDigits = /\d{4}/;

var word = /\*+|+(\s\*?+){1,2}/i;

var noop = function () {

};

function shorten(arr, sLen) {

var newArr = ;

for (var i = 0, len = arr.length; i < len; i++) {

newArr.push(arr.substr(0, sLen));

}

return newArr;

}

function monthUpdate(arrName) {

return function (d, v, i18n) {

var index = i18n.indexOf(v.charAt(0).toUpperCase() + v.substr(1).toLowerCase());

if (~index) {

d.month = index;

}

};

}

function pad(val, len) {

val = String(val);

len = len || 2;

while (val.length < len) {

val = '0' + val;

}

return val;

}

var dayNames = ;

var monthNames = ;

var monthNamesShort = shorten(monthNames, 3);

var dayNamesShort = shorten(dayNames, 3);

fecha.i18n = {

dayNamesShort: dayNamesShort,

dayNames: dayNames,

monthNamesShort: monthNamesShort,

monthNames: monthNames,

amPm: ,

DoFn: function DoFn(D) {

return D + ;

}

};

var formatFlags = {

D: function (dateObj) {

return dateObj.getDay();

},

DD: function (dateObj) {

return pad(dateObj.getDay());

},

Do: function (dateObj, i18n) {

return i18n.DoFn(dateObj.getDate());

},

d: function (dateObj) {

return dateObj.getDate();

},

dd: function (dateObj) {

return pad(dateObj.getDate());

},

ddd: function (dateObj, i18n) {

return i18n.dayNamesShort;

},

dddd: function (dateObj, i18n) {

return i18n.dayNames;

},

M: function (dateObj) {

return dateObj.getMonth() + 1;

},

MM: function (dateObj) {

return pad(dateObj.getMonth() + 1);

},

MMM: function (dateObj, i18n) {

return i18n.monthNamesShort;

},

MMMM: function (dateObj, i18n) {

return i18n.monthNames;

},

yy: function (dateObj) {

return String(dateObj.getFullYear()).substr(2);

},

yyyy: function (dateObj) {

return dateObj.getFullYear();

},

h: function (dateObj) {

return dateObj.getHours() % 12 || 12;

},

hh: function (dateObj) {

return pad(dateObj.getHours() % 12 || 12);

},

H: function (dateObj) {

return dateObj.getHours();

},

HH: function (dateObj) {

return pad(dateObj.getHours());

},

m: function (dateObj) {

return dateObj.getMinutes();

},

mm: function (dateObj) {

return pad(dateObj.getMinutes());

},

s: function (dateObj) {

return dateObj.getSeconds();

},

ss: function (dateObj) {

return pad(dateObj.getSeconds());

},

S: function (dateObj) {

return Math.round(dateObj.getMilliseconds() / 100);

},

SS: function (dateObj) {

return pad(Math.round(dateObj.getMilliseconds() / 10), 2);

},

SSS: function (dateObj) {

return pad(dateObj.getMilliseconds(), 3);

},

a: function (dateObj, i18n) {

return dateObj.getHours() < 12 ? i18n.amPm : i18n.amPm;

},

A: function (dateObj, i18n) {

return dateObj.getHours() < 12 ? i18n.amPm.toUpperCase() : i18n.amPm.toUpperCase();

},

ZZ: function (dateObj) {

var o = dateObj.getTimezoneOffset();

return (o > 0 ? '-' : '+') + pad(Math.floor(Math.abs(o) / 60) \* 100 + Math.abs(o) % 60, 4);

}

};

var parseFlags = {

d: [twoDigits, function (d, v) {

d.day = v;

}],

M: [twoDigits, function (d, v) {

d.month = v - 1;

}],

yy: [twoDigits, function (d, v) {

var da = new Date(), cent = +('' + da.getFullYear()).substr(0, 2);

d.year = '' + (v > 68 ? cent - 1 : cent) + v;

}],

h: [twoDigits, function (d, v) {

d.hour = v;

}],

m: [twoDigits, function (d, v) {

d.minute = v;

}],

s: [twoDigits, function (d, v) {

d.second = v;

}],

yyyy: [fourDigits, function (d, v) {

d.year = v;

}],

S: [/\d/, function (d, v) {

d.millisecond = v \* 100;

}],

SS: [/\d{2}/, function (d, v) {

d.millisecond = v \* 10;

}],

SSS: [threeDigits, function (d, v) {

d.millisecond = v;

}],

D: ,

ddd: ,

MMM: ,

MMMM: ,

a: [word, function (d, v, i18n) {

var val = v.toLowerCase();

if (val === i18n.amPm) {

d.isPm = false;

} else if (val === i18n.amPm) {

d.isPm = true;

}

}],

ZZ: \d\d:?\d\d/, function (d, v) {

var parts = (v + '').match(/(|\d\d)/gi), minutes;

if (parts) {

minutes = +(parts \* 60) + parseInt(parts, 10);

d.timezoneOffset = parts === '+' ? minutes : -minutes;

}

}]

};

parseFlags.DD = parseFlags.DD;

parseFlags.dddd = parseFlags.ddd;

parseFlags.Do = parseFlags.dd = parseFlags.d;

parseFlags.mm = parseFlags.m;

parseFlags.hh = parseFlags.H = parseFlags.HH = parseFlags.h;

parseFlags.MM = parseFlags.M;

parseFlags.ss = parseFlags.s;

parseFlags.A = parseFlags.a;

// Some common format strings

fecha.masks = {

'default': 'ddd MMM dd yyyy HH:mm:ss',

shortDate: 'M/D/yy',

mediumDate: 'MMM d, yyyy',

longDate: 'MMMM d, yyyy',

fullDate: 'dddd, MMMM d, yyyy',

shortTime: 'HH:mm',

mediumTime: 'HH:mm:ss',

longTime: 'HH:mm:ss.SSS'

};

/\*\*\*

\* Format a date

\* @method format

\* @param {Date|number} dateObj

\* @param {string} mask Format of the date, i.e. 'mm-dd-yy' or 'shortDate'

\*/

fecha.format = function (dateObj, mask, i18nSettings) {

var i18n = i18nSettings || fecha.i18n;

if (typeof dateObj === 'number') {

dateObj = new Date(dateObj);

}

if (Object.prototype.toString.call(dateObj) !== '' || isNaN(dateObj.getTime())) {

throw new Error('Invalid Date in fecha.format');

}

mask = fecha.masks || mask || fecha.masks;

return mask.replace(token, function ($0) {

return $0 in formatFlags ? formatFlags(dateObj, i18n) : $0.slice(1, $0.length - 1);

});

};

/\*\*

\* Parse a date string into an object, changes - into /

\* @method parse

\* @param {string} dateStr Date string

\* @param {string} format Date parse format

\* @returns {Date|boolean}

\*/

fecha.parse = function (dateStr, format, i18nSettings) {

var i18n = i18nSettings || fecha.i18n;

if (typeof format !== 'string') {

throw new Error('Invalid format in fecha.parse');

}

format = fecha.masks || format;

// Avoid regular expression denial of service, fail early for really long strings

// https://www.owasp.org/index.php/Regular\_expression\_Denial\_of\_Service\_-\_ReDoS

if (dateStr.length > 1000) {

return false;

}

var isValid = true;

var dateInfo = {};

format.replace(token, function ($0) {

if (parseFlags) {

var info = parseFlags;

var index = dateStr.search(info);

if (!~index) {

isValid = false;

} else {

dateStr.replace(info, function (result) {

info(dateInfo, result, i18n);

dateStr = dateStr.substr(index + result.length);

return result;

});

}

}

return parseFlags ? '' : $0.slice(1, $0.length - 1);

});

if (!isValid) {

return false;

}

var today = new Date();

if (dateInfo.isPm === true && dateInfo.hour != null && +dateInfo.hour !== 12) {

dateInfo.hour = +dateInfo.hour + 12;

} else if (dateInfo.isPm === false && +dateInfo.hour === 12) {

dateInfo.hour = 0;

}

var date;

if (dateInfo.timezoneOffset != null) {

dateInfo.minute = +(dateInfo.minute || 0) - +dateInfo.timezoneOffset;

date = new Date(Date.UTC(dateInfo.year || today.getFullYear(), dateInfo.month || 0, dateInfo.day || 1,

dateInfo.hour || 0, dateInfo.minute || 0, dateInfo.second || 0, dateInfo.millisecond || 0));

} else {

date = new Date(dateInfo.year || today.getFullYear(), dateInfo.month || 0, dateInfo.day || 1,

dateInfo.hour || 0, dateInfo.minute || 0, dateInfo.second || 0, dateInfo.millisecond || 0);

}

return date;

};

/\* istanbul ignore next \*/

if (typeof module !== 'undefined' && module.exports) {

module.exports = fecha;

} else if (typeof define === 'function' && define.amd) {

define(function () {

return fecha;

});

} else {

main.fecha = fecha;

}

})(this);

export default {

computed: {

device() {

return this.$store.state.app.device

}

},

mounted() {

// In order to fix the click on menu on the ios device will trigger the mouseleave bug

// https://github.com/PanJiaChen/vue-element-admin/issues/1135

this.fixBugIniOS()

},

methods: {

fixBugIniOS() {

const $subMenu = this.$refs.subMenu

if ($subMenu) {

const handleMouseleave = $subMenu.handleMouseleave

$subMenu.handleMouseleave = (e) => {

if (this.device === 'mobile') {

return

}

handleMouseleave(e)

}

}

}

}

}

import defaultSettings from '@/settings'

const title = defaultSettings.title || 'Vue Admin Template'

export default function getPageTitle(pageTitle) {

if (pageTitle) {

return `${pageTitle} - ${title}`

}

return `${title}`

}

const getters = {

device: state => state.app.device,

permission\_routes: state => state.permission.routes,

isMobile: state => state.state.app.isMobile

}

export default getters

import {Message} from 'element-ui'

import axios from 'axios'

axios.defaults.withCredentials=true;

axios.defaults.baseURL = process.env.VUE\_APP\_API

axios.defaults.timeout = 30 \* 1000 // 设置接口响应时间

/\*\*

\* 功能描述：Http Request 拦截器

\*/

axios.interceptors.request.use((config) => {

return config

}, (error) => { // 请求错误时做些事(接口错误、超时等)

Message.error(`请求参数错误`, 10)

return Promise.reject(error) // 在调用的那边可以拿到(catch)你想返回的错误信息

})

/\*\*

\* 功能描述：Http Response 拦截器

\*/

axios.interceptors.response.use((res) => {

let isTips = res.config.showMessage === undefined || res.config.showMessage === true

let data = res.data

/\* 根据返回的code值来做不同的处理（和后端约定） \*/

switch (data.code) {

case 200:

return Promise.resolve(data)

default:

isTips && Message.error(`服务器返回异常：${data.message}`, 10)

return Promise.reject(res)

}

}, (error) => {

Message.error(`服务器返回异常：${error}`, 10)

if (error && error.response) {

switch (error.response.status) {

case 400:

break

default:

break

}

}

return Promise.reject(error)

})

export default {

get (url, data = {}) {

return new Promise((resolve, reject) => {

axios.get(url, data).then(response => {

resolve(response)

}).catch(err => {

reject(err)

})

})

},

delete (url, data = {}) {

return new Promise((resolve, reject) => {

axios.delete(url, data).then(response => {

resolve(response)

}).catch(err => {

reject(err)

})

})

},

post (url, data = {}) {

return new Promise((resolve, reject) => {

axios.post(url, data).then(response => {

resolve(response)

}, err => {

reject(err)

})

})

},

put (url, data = {}) {

return new Promise((resolve, reject) => {

axios.put(url, data).then(response => {

resolve(response)

}, err => {

reject(err)

})

})

},

patch (url, data = {}) {

return new Promise((resolve, reject) => {

axios.patch(url, data).then(response => {

resolve(response)

}, err => {

reject(err)

})

})

}

}

export { default as Sidebar } from './Sidebar'

export { default as AppMain } from './AppMain'

export { default as EndPart } from './EndPart'

import Vue from 'vue'

import Router from 'vue-router'

Vue.use(Router)

/\* Layout \*/

import Layout from '@/layout'

export const constantRoutes = [

{

path: '/redirect',

component: Layout,

hidden: true,

children: [

{

path: '/redirect/:path\*',

component: () => import('@/views/redirect/index')

}

]

},

{

path: '/404',

component: () => import('@/views/404'),

hidden: true

},

{

path: '/',

component: Layout,

redirect: '/home',

children: [{

path: 'home',

name: 'Home',

component: () => import('@/views/home/index'),

meta: { title: '主页', icon: 'dashboard', affix: true }

}]

},

{

path: '/article',

component: Layout,

children: [

{

path: '',

name: 'web\_article',

component: () => import('@/views/article/index'),

meta: { title: '文章', icon: 'article' },

},

{

path: ':id(\\d+)',

name: 'web\_article\_detail',

component: () => import('@/views/article/detail'),

meta: { title: '文章详情', noCache: true, activeMenu: '/article/index', icon: 'article' },

hidden: true

}

]

},

{

path: '/video',

component: Layout,

children: [

{

path: '',

name: 'web\_video',

component: () => import('@/views/video/index'),

meta: { title: '视频', icon: 'video' },

},

{

path: ':id(\\d+)',

name: 'web\_video\_detail',

component: () => import('@/views/video/detail'),

meta: { title: '视频详情', noCache: true, activeMenu: '/video/index', icon: 'video' },

hidden: true

}

]

},

{

path: '/company',

component: Layout,

children: [{

path: '',

name: 'web\_company',

component: () => import('@/views/company/index'),

meta: { title: '公司信息', icon: 'subordinate' },

}]

}

]

const createRouter = () => new Router({

// mode: 'history', // require service support

routes: constantRoutes

})

const router = createRouter()

// Detail see: https://github.com/vuejs/vue-router/issues/1234#issuecomment-357941465

export function resetRouter() {

const newRouter = createRouter()

router.matcher = newRouter.matcher // reset router

}

export default router

import Vue from 'vue'

import Vuex from 'vuex'

import getters from './getters'

Vue.use(Vuex)

// https://webpack.js.org/guides/dependency-management/#requirecontext

const modulesFiles = require.context('./modules', true, /\.js$/)

// you do not need `import app from './modules/app'`

// it will auto require all vuex module from modules file

const modules = modulesFiles.keys().reduce((modules, modulePath) => {

// set './app.js' => 'app'

const moduleName = modulePath.replace(/^\.\/(.\*)\.\w+$/, '$1')

const value = modulesFiles(modulePath)

modules = value.default

return modules

}, {})

const store = new Vuex.Store({

modules,

getters

})

export default store

/\*\*

\* Created by PanJiaChen on 16/11/18.

\*/

/\*\*

\* Parse the time to string

\* @param {(Object|string|number)} time

\* @param {string} cFormat

\* @returns {string | null}

\*/

export function parseTime(time, cFormat) {

if (arguments.length === 0) {

return null

}

const format = cFormat || '{y}-{m}-{d} {h}:{i}:{s}'

let date

if (typeof time === 'object') {

date = time

} else {

if ((typeof time === 'string') && (/^+$/.test(time))) {

time = parseInt(time)

}

if ((typeof time === 'number') && (time.toString().length === 10)) {

time = time \* 1000

}

date = new Date(time)

}

const formatObj = {

y: date.getFullYear(),

m: date.getMonth() + 1,

d: date.getDate(),

h: date.getHours(),

i: date.getMinutes(),

s: date.getSeconds(),

a: date.getDay()

}

const time\_str = format.replace(/{()+}/g, (result, key) => {

const value = formatObj

// Note: getDay() returns 0 on Sunday

if (key === 'a') { return }

return value.toString().padStart(2, '0')

})

return time\_str

}

/\*\*

\* @param {number} time

\* @param {string} option

\* @returns {string}

\*/

export function formatTime(time, option) {

if (('' + time).length === 10) {

time = parseInt(time) \* 1000

} else {

time = +time

}

const d = new Date(time)

const now = Date.now()

const diff = (now - d) / 1000

if (diff < 30) {

return '刚刚'

} else if (diff < 3600) {

// less 1 hour

return Math.ceil(diff / 60) + '分钟前'

} else if (diff < 3600 \* 24) {

return Math.ceil(diff / 3600) + '小时前'

} else if (diff < 3600 \* 24 \* 2) {

return '1天前'

}

if (option) {

return parseTime(time, option)

} else {

return (

d.getMonth() +

1 +

'月' +

d.getDate() +

'日' +

d.getHours() +

'时' +

d.getMinutes() +

'分'

)

}

}

/\*\*

\* @param {string} url

\* @returns {Object}

\*/

export function param2Obj(url) {

const search = url.split('?')

if (!search) {

return {}

}

return JSON.parse(

'{"' +

decodeURIComponent(search)

.replace(/"/g, '\\"')

.replace(/&/g, '","')

.replace(/=/g, '":"')

.replace(/\+/g, ' ') +

'"}'

)

}

import Vue from 'vue'

import SvgIcon from '@/components/SvgIcon'// svg component

// register globally

Vue.component('svg-icon', SvgIcon)

const req = require.context('./svg', false, /\.svg$/)

const requireAll = requireContext => requireContext.keys().map(requireContext)

requireAll(req)

import Vue from 'vue'

import 'normalize.css/normalize.css' // CSS重置的现代替代方法

import ElementUI from 'element-ui'

import 'element-ui/lib/theme-chalk/index.css'

import '@/styles/index.scss' // global css

import App from './App.vue'

import store from './store'

import router from './router'

import '@/icons' // 图标

import VueAwesomeSwiper from 'vue-awesome-swiper'

// import style

import 'swiper/css/swiper.css'

Vue.use(VueAwesomeSwiper)

import getPageTitle from '@/utils/get-page-title'

// import 'lib-flexible'

// import '@/utils/rem'

import 'lib-flexible/flexible'

Vue.use(ElementUI)

Vue.config.productionTip = false

router.beforeEach(async(to, from, next) => {

document.title = getPageTitle(to.meta.title)

next()

})

new Vue({

el: '#app',

router,

store,

render: h => h(App),

})

import { constantRoutes } from '@/router'

/\*\*

\* Use meta.role to determine if the current user has permission

\* @param roles

\* @param route

\*/

const state = {

routes: constantRoutes,

}

export default {

namespaced: true,

state,

}

// 设置 rem 函数

function setRem () {

// 320 默认大小16px; 320px = 20rem ;每个元素px基础上/16

let htmlWidth = document.documentElement.clientWidth || document.body.clientWidth;

//得到html的Dom元素

let htmlDom = document.getElementsByTagName('html');

//设置根元素字体大小

htmlDom.style.fontSize= htmlWidth/10 + 'px';

}

// 初始化

setRem();

// 改变窗口大小时重新设置 rem

window.onresize = function () {

setRem()

}

import store from '@/store'

const { body } = document

const WIDTH = 992 // refer to Bootstrap's responsive design

export default {

watch: {

$route(route) {

if (this.device === 'mobile' && this.sidebar.opened) {

store.dispatch('app/closeSideBar', { withoutAnimation: false })

}

}

},

beforeMount() {

window.addEventListener('resize', this.$\_resizeHandler)

},

beforeDestroy() {

window.removeEventListener('resize', this.$\_resizeHandler)

},

mounted() {

const isMobile = this.$\_isMobile()

if (isMobile) {

store.dispatch('app/toggleDevice', 'mobile')

store.dispatch('app/closeSideBar', { withoutAnimation: true })

}

},

methods: {

// use $\_ for mixins properties

// https://vuejs.org/v2/style-guide/index.html#Private-property-names-essential

$\_isMobile() {

const rect = body.getBoundingClientRect()

return rect.width - 1 < WIDTH

},

$\_resizeHandler() {

if (!document.hidden) {

const isMobile = this.$\_isMobile()

store.dispatch('app/toggleDevice', isMobile ? 'mobile' : 'desktop')

if (isMobile) {

store.dispatch('app/closeSideBar', { withoutAnimation: true })

}

}

}

}

}

Math.easeInOutQuad = function(t, b, c, d) {

t /= d / 2

if (t < 1) {

return c / 2 \* t \* t + b

}

t--

return -c / 2 \* (t \* (t - 2) - 1) + b

}

// requestAnimationFrame for Smart Animating http://goo.gl/sx5sts

var requestAnimFrame = (function() {

return window.requestAnimationFrame || window.webkitRequestAnimationFrame || window.mozRequestAnimationFrame || function(callback) { window.setTimeout(callback, 1000 / 60) }

})()

/\*\*

\* Because it's so fucking difficult to detect the scrolling element, just move them all

\* @param {number} amount

\*/

function move(amount) {

document.documentElement.scrollTop = amount

document.body.parentNode.scrollTop = amount

document.body.scrollTop = amount

}

function position() {

return document.documentElement.scrollTop || document.body.parentNode.scrollTop || document.body.scrollTop

}

/\*\*

\* @param {number} to

\* @param {number} duration

\* @param {Function} callback

\*/

export function scrollTo(to, duration, callback) {

const start = position()

const change = to - start

const increment = 20

let currentTime = 0

duration = (typeof (duration) === 'undefined') ? 500 : duration

var animateScroll = function() {

// increment the time

currentTime += increment

// find the value with the quadratic in-out easing function

var val = Math.easeInOutQuad(currentTime, start, change, duration)

// move the document.body

move(val)

// do the animation unless its over

if (currentTime < duration) {

requestAnimFrame(animateScroll)

} else {

if (callback && typeof (callback) === 'function') {

// the animation is done so lets callback

callback()

}

}

}

animateScroll()

}

module.exports = {

title: '广东米果传媒有限公司',

/\*\*

\* @type {boolean} true | false

\* @description Whether show the settings right-panel

\*/

showSettings: true,

/\*\*

\* @type {boolean} true | false

\* @description Whether need tagsView

\*/

tagsView: true,

/\*\*

\* @type {boolean} true | false

\* @description Whether fix the header

\*/

fixedHeader: false,

/\*\*

\* @type {boolean} true | false

\* @description Whether show the logo in sidebar

\*/

sidebarLogo: false,

/\*\*

\* @type {string | array} 'production' |

\* @description Need show err logs component.

\* The default is only used in the production env

\* If you want to also use it in dev, you can pass

\*/

errorLog: 'production'

}

export default {

/\*\*

\* 功能描述：计算文本的长度

\* @param strKit

\* @returns {number}

\*/

getByteLen (strKit) {

let len = 0

if (typeof strKit === 'string' && strKit.constructor === String) {

for (let i = 0; i < strKit.length; i++) {

if (strKit.match(//ig)) { // 全角

len += 2

} else {

len += 1

}

}

}

return len

}

}

export function formatDate(fmt,time) {

const a = new Date()

const o = {

'M+': time.getMonth() + 1, // 月份

'd+': time.getDate(), // 日

'h+': time.getHours(), // 小时

'm+': time.getMinutes(), // 分

's+': time.getSeconds(), // 秒

'q+': Math.floor((time.getMonth() + 3) / 3), // 季度

'S': time.getMilliseconds() // 毫秒

};

if (/(y+)/.test(fmt)) {

fmt = fmt.replace(RegExp.$1, (time.getFullYear() + '').substr(4 - RegExp.$1.length));

}

for (const k in o) {

if (new RegExp('(' + k + ')').test(fmt)) {

fmt = fmt.replace(RegExp.$1, (RegExp.$1.length === 1) ?

(o) : (('00' + o).substr(('' + o).length)));

}

}

return fmt;

};

/\*\*

\* Created by PanJiaChen on 16/11/18.

\*/

/\*\*

\* @param {string} path

\* @returns {Boolean}

\*/

export function isExternal(path) {

return /^(https?:|mailto:|tel:)/.test(path)

}

import HttpKit from '@/utils/http-kit'

export default {

findAllSwiper () {

return HttpKit.post(`/client/swiper/find\_all\_exist`, "a").then(

res => res.data

)

},

findSomeArticle (form) {

return HttpKit.post(`/client/article/find\_some`, form).then(

res => res.data

)

},

findAllArticle (form) {

return HttpKit.post(`/client/article/find\_all\_by\_keywords`, form).then(

res => res.data

)

},

findOneArticle (id) {

return HttpKit.get(`/client/article/find\_one\_by\_id/${id}`).then(

res => res.data

)

},

findAllVideo (form) {

return HttpKit.post(`/client/video/find\_all\_by\_keywords`, form).then(

res => res.data

)

},

findOneVideo (id) {

return HttpKit.get(`/client/video/find\_one\_by\_id/${id}`).then(

res => res.data

)

},

getBilibili (id) {

return HttpKit.get(`/bilibili?aid=${id}&cid=21890669&page=1&high\_quality=1&danmaku=0`).then(

res => res

)

}

}

管理后台前端代码

<template>

<div class="wscn-http404-container">

<div class="wscn-http404">

<div class="pic-404">

<img class="pic-404\_\_parent" src="@/assets/404\_images/404.png" alt="404">

<img class="pic-404\_\_child left" src="@/assets/404\_images/404\_cloud.png" alt="404">

<img class="pic-404\_\_child mid" src="@/assets/404\_images/404\_cloud.png" alt="404">

<img class="pic-404\_\_child right" src="@/assets/404\_images/404\_cloud.png" alt="404">

</div>

<div class="bullshit">

<div class="bullshit\_\_oops">OOPS!</div>

<div class="bullshit\_\_info">All rights reserved

<a style="color:#20a0ff" href="https://wallstreetcn.com" target="\_blank">wallstreetcn</a>

</div>

<div class="bullshit\_\_headline">{{ message }}</div>

<div class="bullshit\_\_info">Please check that the URL you entered is correct, or click the button below to return to the homepage.</div>

<a href="" class="bullshit\_\_return-home">Back to home</a>

</div>

</div>

</div>

</template>

<script>

export default {

name: 'Page404',

computed: {

message() {

return 'The webmaster said that you can not enter this page...'

}

}

}

</script>

<style lang="scss" scoped>

.wscn-http404-container{

transform: translate(-50%,-50%);

position: absolute;

top: 40%;

left: 50%;

}

.wscn-http404 {

position: relative;

width: 1200px;

padding: 0 50px;

overflow: hidden;

.pic-404 {

position: relative;

float: left;

width: 600px;

overflow: hidden;

&\_\_parent {

width: 100%;

}

&\_\_child {

position: absolute;

&.left {

width: 80px;

top: 17px;

left: 220px;

opacity: 0;

animation-name: cloudLeft;

animation-duration: 2s;

animation-timing-function: linear;

animation-fill-mode: forwards;

animation-delay: 1s;

}

&.mid {

width: 46px;

top: 10px;

left: 420px;

opacity: 0;

animation-name: cloudMid;

animation-duration: 2s;

animation-timing-function: linear;

animation-fill-mode: forwards;

animation-delay: 1.2s;

}

&.right {

width: 62px;

top: 100px;

left: 500px;

opacity: 0;

animation-name: cloudRight;

animation-duration: 2s;

animation-timing-function: linear;

animation-fill-mode: forwards;

animation-delay: 1s;

}

@keyframes cloudLeft {

0% {

top: 17px;

left: 220px;

opacity: 0;

}

20% {

top: 33px;

left: 188px;

opacity: 1;

}

80% {

top: 81px;

left: 92px;

opacity: 1;

}

100% {

top: 97px;

left: 60px;

opacity: 0;

}

}

@keyframes cloudMid {

0% {

top: 10px;

left: 420px;

opacity: 0;

}

20% {

top: 40px;

left: 360px;

opacity: 1;

}

70% {

top: 130px;

left: 180px;

opacity: 1;

}

100% {

top: 160px;

left: 120px;

opacity: 0;

}

}

@keyframes cloudRight {

0% {

top: 100px;

left: 500px;

opacity: 0;

}

20% {

top: 120px;

left: 460px;

opacity: 1;

}

80% {

top: 180px;

left: 340px;

opacity: 1;

}

100% {

top: 200px;

left: 300px;

opacity: 0;

}

}

}

}

.bullshit {

position: relative;

float: left;

width: 300px;

padding: 30px 0;

overflow: hidden;

&\_\_oops {

font-size: 32px;

font-weight: bold;

line-height: 40px;

color: #1482f0;

opacity: 0;

margin-bottom: 20px;

animation-name: slideUp;

animation-duration: 0.5s;

animation-fill-mode: forwards;

}

&\_\_headline {

font-size: 20px;

line-height: 24px;

color: #222;

font-weight: bold;

opacity: 0;

margin-bottom: 10px;

animation-name: slideUp;

animation-duration: 0.5s;

animation-delay: 0.1s;

animation-fill-mode: forwards;

}

&\_\_info {

font-size: 13px;

line-height: 21px;

color: grey;

opacity: 0;

margin-bottom: 30px;

animation-name: slideUp;

animation-duration: 0.5s;

animation-delay: 0.2s;

animation-fill-mode: forwards;

}

&\_\_return-home {

display: block;

float: left;

width: 110px;

height: 36px;

background: #1482f0;

border-radius: 100px;

text-align: center;

color: #ffffff;

opacity: 0;

font-size: 14px;

line-height: 36px;

cursor: pointer;

animation-name: slideUp;

animation-duration: 0.5s;

animation-delay: 0.3s;

animation-fill-mode: forwards;

}

@keyframes slideUp {

0% {

transform: translateY(60px);

opacity: 0;

}

100% {

transform: translateY(0);

opacity: 1;

}

}

}

}

</style>

<template>

<div id="app">

<router-view />

</div>

</template>

<script>

export default {

name: 'App'

}

</script>

<template>

<section class="app-main">

<transition name="fade-transform" mode="out-in">

<router-view :key="key" />

</transition>

</section>

</template>

<script>

export default {

name: 'AppMain',

computed: {

key() {

return this.$route.path

}

}

}

</script>

<style scoped>

.app-main {

/\*50 = navbar \*/

min-height: calc(100vh - 50px);

width: 100%;

position: relative;

overflow: hidden;

}

.fixed-header+.app-main {

padding-top: 50px;

}

</style>

<style lang="scss">

// fix css style bug in open el-dialog

.el-popup-parent--hidden {

.fixed-header {

padding-right: 15px;

}

}

</style>

<template>

<div class="createPost-container">

<el-form ref="postForm" :model="postForm" :rules="rules" class="form-container">

<sticky :z-index="10" :class-name="'sub-navbar '+postForm.status">

<el-button v-loading="loading" style="margin-left: 10px;" type="success" @click="submitForm(1)">

送审

</el-button>

<el-button v-loading="loading" type="warning" @click="submitForm(3)">

保存草稿

</el-button>

<el-button v-loading="loading" type="warning" @click="submitForm(2)">

保存并返回上一层

</el-button>

</sticky>

<div class="createPost-main-container">

<el-row>

<el-col :span="24">

<el-form-item style="margin-bottom: 40px;" prop="title">

<material-input v-model="postForm.title">

标题

</material-input>

</el-form-item>

<div class="postInfo-container">

<el-row>

<el-col :span="8">

<el-form-item label-width="60px" label="作者:" class="postInfo-container-item">

<el-input v-model="postForm.author" />

</el-form-item>

</el-col>

<el-col :span="3">

<el-button v-show="!postForm.pic" type="primary" size="mini" @click="imageCropperShow()">

上传图片

</el-button>

<el-button v-show="postForm.pic" type="primary" size="mini" @click="imageCropperShow()">

更新图片

</el-button>

<el-button v-show="postForm.pic" type="primary" size="mini" @click="picShow(postForm.pic)">

预览

</el-button>

</el-col>

</el-row>

</div>

</el-col>

</el-row>

<el-form-item prop="content" style="margin-bottom: 30px;">

<Editor

:id="postForm.id"

ref="editor"

v-model="postForm.article"

:init="editorInit"

:disabled="isEditorDisabled"

@input="handleInput"

@onClick="handleClick"

:height="400"

/>

<div class="ext-btn" style="margin-top: 30px; float: right">

<el-button type="danger" @click="clearClick">清空内容</el-button>

<el-button @click="disableClick">{{ !isEditorDisabled ? '禁用' : '启用' }}</el-button>

</div>

</el-form-item>

</div>

</el-form>

<my-upload

method="POST"

field="file"

v-model="cropperShow"

:width=320

:height=240

:url="this.$store.state.settings.uploadUrl"

lang-type='zh'

img-format='jpg'

img-bgc='#FFF'

:no-circle=true

@crop-upload-success="cropUploadSuccess">

</my-upload>

<el-dialog title="图片预览" :visible.sync="picVisible" width="400px" center>

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

<img :src="answerPicImageUrl" alt="">

</div>

<span slot="footer" class="dialog-footer">

<el-button @click="picVisible = false">确定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import axios from 'axios'

// 引入组件

import tinymce from 'tinymce/tinymce'

import Editor from '@tinymce/tinymce-vue'

// 引入富文本编辑器主题的js和css

import 'tinymce/themes/silver/theme.min.js'

import 'tinymce/skins/ui/oxide/skin.min.css'

// 扩展插件

import 'tinymce/plugins/image'

import 'tinymce/plugins/link'

import 'tinymce/plugins/code'

import 'tinymce/plugins/table'

import 'tinymce/plugins/lists'

import 'tinymce/plugins/paste'

// 字数统计插件

import 'tinymce/plugins/wordcount'

import MaterialInput from '@/components/MaterialInput'

import Sticky from '@/components/Sticky' // 粘性header组件

import WebArticleApi from "@/api/website/WebArticleApi";

import 'babel-polyfill'; // es6 shim

import myUpload from 'vue-image-crop-upload';

const defaultForm = {

id: undefined,

status: '',

title: '', // 文章题目

article: '', // 文章内容

author: '', // 文章作者

pic: '' // 文章头图

}

export default {

components: { Editor,MaterialInput,Sticky,myUpload },

props: {

isEdit: {

type: Boolean,

default: false

},

plugins: {

type: ,

default: 'link lists image code table wordcount paste'

},

toolbar: {

type: ,

default: 'bold italic underline strikethrough | fontsizeselect | forecolor backcolor | alignleft aligncenter alignright alignjustify | bullist numlist | outdent indent blockquote | undo redo | link unlink image code | removeformat'

},

menubar: {

type: String,

default: 'file edit insert view format table'

},

height: {

type: ,

required: false,

default: 360

},

width: {

type: ,

required: false,

default: 'auto'

},

},

accept: {

default: 'image/jpeg, image/png, image/jpg, image/gif',

type: String

},

data() {

const validateRequire = (rule, value, callback) => {

if (value === '') {

this.$message({

message: rule.field + '为必传项',

type: 'error'

})

callback(new Error(rule.field + '为必传项'))

} else {

callback()

}

}

return {

answerPicImageUrl: "",

picVisible: false,

cropperShow: false,

\_isEdit: true,

type: ,

status: '',

postForm: Object.assign({}, defaultForm),

loading: false,

isEditorDisabled: false,

rules: {

title: ,

content: ,

},

tempRoute: {},

editorInit: {

paste\_data\_images: true,

language\_url: './static/tinymce/langs/zh\_CN.js',

language: 'zh\_CN',

skin\_url: './static/tinymce/skins/ui/oxide',

height: 900,

width: 800,

plugins: this.plugins,

toolbar: this.toolbar,

statusbar: true, // 底部的状态栏

menubar: true, // 最上方的菜单

branding: false, // 水印“Powered by TinyMCE”

images\_upload\_handler: (blobInfo, success, failure) => {

// const img = 'data:image/png;base64,' + blobInfo.base64()

// success(img)

// this.$emit('imgUpload', blobInfo, success, failure)

if(blobInfo.size / 1024 / 1024 < 8){

failure('文件体积过大')

}

else if(!this.type.includes(blobInfo.blob().type)) {

failure('文件格式无效')

console.log('文件格式无效')

}

else{

console.log('上传')

let formData = new FormData()

formData.append("file",blobInfo.blob())

axios.post(this.$store.state.settings.uploadUrl,formData)

.then((res) => {

console.log(res)

if (res.code === 200) {

console.log(this.$store.state.settings.callbackUrl+res.data)

success(this.$store.state.settings.callbackUrl+res.data)

} else {

failure('上传失败!')

}

})

}

}

},

}

},

created() {

this.\_isEdit = this.isEdit

if (this.isEdit) {

const id = this.$route.params && this.$route.params.id

this.fetchData(id)

}

// Why need to make a copy of this.$route here?

// Because if you enter this page and quickly switch tag, may be in the execution of the setTagsViewTitle function, this.$route is no longer pointing to the current page

// https://github.com/PanJiaChen/vue-element-admin/issues/1221

this.tempRoute = Object.assign({}, this.$route)

},

mounted() {

tinymce.init({

})

},

methods: {

fetchData(id) {

WebArticleApi.findOne(id).then(data => {

this.postForm = data

this.setPageTitle()

}).catch(err => {

console.log(err)

})

},

setPageTitle() {

const title = '编辑文章'

document.title = `${title} : ${this.postForm.id}`

},

submitForm(value) {

this.$refs.postForm.validate((valid) => {

if (valid) {

this.loading = true

console.log(this.isEdit)

if(this.\_isEdit){

WebArticleApi.update(this.postForm)

}

else{

WebArticleApi.add(this.postForm).then(data => {

console.log(data)

this.postForm.id = data

this.setPageTitle()

}).catch(err => {

console.log(err)

})

this.\_isEdit = true

}

if(value === 1){

this.$notify({

title: '成功',

message: '发布文章成功',

type: 'success',

duration: 2000

})

this.postForm.status = 'reviewing'

this.loading = false

if(this.isEdit){

this.$router.push({path:'../list'})

}

else{

this.$router.push({path:'./list'})

}

}

else if(value === 2) {

this.$notify({

title: '成功',

message: '更新成功',

type: 'success',

duration: 2000

})

this.postForm.status = 'draft'

this.loading = false

if(this.isEdit){

this.$router.push({path:'../list'})

}

else{

this.$router.push({path:'./list'})

}

}

else {

this.$notify({

title: '成功',

message: '更新成功',

type: 'success',

duration: 2000

})

this.postForm.status = 'draft'

this.loading = false

}

} else {

console.log('error submit!!')

return false

}

})

},

// 显示图片上传模块

imageCropperShow() {

this.cropperShow = !this.cropperShow

},

// 图片上传成功后执行

cropUploadSuccess(jsonData, field){

this.postForm.pic = jsonData.data

const tempData = Object.assign({}, this.postForm)

WebArticleApi.update(tempData).then(() => {

this.$notify({

title: 'success',

message: '图片上传成功',

type: 'success',

duration: 2000

})

this.getList()

})

},

picShow(pic) {

this.picVisible = !this.picVisible

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + pic

},

// 输入事件

handleInput(value) {

// console.log(value)

},

// 点击事件

handleClick(e, editor) {

// console.log(e, editor)

},

// 清空事件

clearClick() {

this.postForm.article = ''

},

// 禁用事件

disableClick() {

this.isEditorDisabled = !this.isEditorDisabled

}

}

}

</script>

<style lang="scss" scoped>

@import "~@/styles/mixin.scss";

.createPost-container {

position: relative;

.createPost-main-container {

padding: 40px 45px 20px 50px;

.postInfo-container {

position: relative;

@include clearfix;

margin-bottom: 10px;

.postInfo-container-item {

float: left;

}

}

}

.word-counter {

width: 40px;

position: absolute;

right: 10px;

top: 0px;

}

}

.article-textarea /deep/ {

textarea {

padding-right: 40px;

resize: none;

border: none;

border-radius: 0px;

border-bottom: 1px solid #bfcbd9;

}

}

</style>

<template>

<div class="createPost-container">

<el-form ref="postForm" :model="postForm" :rules="rules" class="form-container">

<sticky :z-index="10" :class-name="'sub-navbar '+postForm.status">

<el-button v-loading="loading" style="margin-left: 10px;" type="success" @click="submitForm(1)">

过审

</el-button>

<el-button v-loading="loading" style="margin-left: 10px;" type="success" @click="submitForm(3)">

待定

</el-button>

<el-button v-loading="loading" type="warning" @click="submitForm(2)">

驳回

</el-button>

</sticky>

<div class="createPost-main-container">

<el-row>

<el-col :span="24">

<el-form-item style="margin-bottom: 40px;" prop="title">

<material-input v-model="postForm.title">

标题

</material-input>

</el-form-item>

<div class="postInfo-container">

<el-row>

<el-col :span="6">

<el-form-item label-width="60px" label="作者:" class="postInfo-container-item">

<el-input v-model="postForm.author" />

</el-form-item>

</el-col>

<el-col :span="6">

<el-form-item label-width="60px" label="审阅人:" class="postInfo-container-item">

<el-input v-model="postForm.reviewer" />

</el-form-item>

</el-col>

<el-col :span="4">

<el-button v-show="!postForm.pic" type="primary" size="mini" @click="imageCropperShow()">

上传图片

</el-button>

<el-button v-show="postForm.pic" type="primary" size="mini" @click="imageCropperShow()">

更新图片

</el-button>

<el-button v-show="postForm.pic" type="primary" size="mini" @click="picShow(postForm.pic)">

预览

</el-button>

</el-col>

</el-row>

</div>

</el-col>

</el-row>

<!-- <div class="editor\_container">-->

<el-form-item prop="content">

<Editor

style="text-align: center"

:id="postForm.id"

ref="editor"

v-model="postForm.article"

:init="editorInit"

:disabled="isEditorDisabled"

@input="handleInput"

@onClick="handleClick"

/>

<div class="ext-btn" style="margin-top: 30px; float: right">

<el-button type="danger" @click="clearClick">清空内容</el-button>

<el-button @click="disableClick">{{ !isEditorDisabled ? '禁用' : '启用' }}</el-button>

</div>

</el-form-item>

<!-- </div>-->

</div>

</el-form>

<my-upload

method="POST"

field="file"

v-model="cropperShow"

:width=320

:height=240

:url="this.$store.state.settings.uploadUrl"

lang-type='zh'

img-format='jpg'

img-bgc='#FFF'

:no-circle=true

@crop-upload-success="cropUploadSuccess">

</my-upload>

<el-dialog title="图片预览" :visible.sync="picVisible" width="400px" center>

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

<img :src="answerPicImageUrl" alt="">

</div>

<span slot="footer" class="dialog-footer">

<el-button @click="picVisible = false">确定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import axios from 'axios'

// 引入组件

import tinymce from 'tinymce/tinymce'

import Editor from '@tinymce/tinymce-vue'

// 引入富文本编辑器主题的js和css

import 'tinymce/themes/silver/theme.min.js'

import 'tinymce/skins/ui/oxide/skin.min.css'

// 扩展插件

import 'tinymce/plugins/image'

import 'tinymce/plugins/link'

import 'tinymce/plugins/code'

import 'tinymce/plugins/table'

import 'tinymce/plugins/lists'

import 'tinymce/plugins/paste'

import 'tinymce/plugins/hr'

// import 'tinymce/plugins/emoticons'

import 'tinymce/plugins/fullscreen'

import './PasteUpload/index'

// 字数统计插件

import 'tinymce/plugins/wordcount'

import MaterialInput from '@/components/MaterialInput'

import Sticky from '@/components/Sticky' // 粘性header组件

import WebArticleApi from "@/api/website/WebArticleApi";

import 'babel-polyfill'; // es6 shim

import myUpload from 'vue-image-crop-upload';

const defaultForm = {

id: undefined,

status: '',

title: '', // 文章题目

article: '', // 文章内容

author: '', // 文章作者

reviewer: '', // 文章审核人

pic: '' // 文章头图

}

export default {

components: { Editor,MaterialInput,Sticky,myUpload },

props: {

isEdit: {

type: Boolean,

default: false

},

plugins: {

type: ,

default: 'link lists image code table wordcount print paste paste-upload preview fullscreen media template code codesample table charmap hr pagebreak nonbreaking anchor insertdatetime advlist imagetools textpattern help emoticons autosave bdmap indent2em autoresize lineheight formatpainter axupimgs'

},

toolbar: {

type: ,

default: 'bold italic underline strikethrough paste-upload | fontsizeselect | forecolor backcolor | alignleft aligncenter alignright alignjustify | bullist numlist | outdent indent blockquote | undo redo | link unlink image code | removeformat hr fullscreen'

},

menubar: {

type: String,

default: 'file edit insert view format table'

},

height: {

type: ,

required: false,

default: 360

},

width: {

type: ,

required: false,

default: 'auto'

},

},

accept: {

default: 'image/jpeg, image/png, image/jpg, image/gif',

type: String

},

data() {

const validateRequire = (rule, value, callback) => {

if (value === '') {

this.$message({

message: rule.field + '为必传项',

type: 'error'

})

callback(new Error(rule.field + '为必传项'))

} else {

callback()

}

}

return {

answerPicImageUrl: "",

picVisible: false,

cropperShow: false,

\_isEdit: true,

type: ,

status: '',

postForm: Object.assign({}, defaultForm),

loading: false,

isEditorDisabled: false,

rules: {

title: ,

content: ,

},

tempRoute: {},

editorInit: {

paste\_data\_images: true,

language\_url: './static/tinymce/langs/zh\_CN.js',

language: 'zh\_CN',

skin\_url: './static/tinymce/skins/ui/oxide',

height: 800,

width: 800,

plugins: this.plugins,

toolbar: this.toolbar,

statusbar: true, // 底部的状态栏

menubar: true, // 最上方的菜单

branding: false, // 水印“Powered by TinyMCE”

pasteUploadConfig: {

url: this.$store.state.settings.uploadUrl,

diyExtractUrl: true,

extractUrlFun: (res) => {

// console.log(res.data)

return this.$store.state.settings.callbackUrl + res.data + '';

}

},

images\_upload\_handler: (blobInfo, success, failure) => {

// const img = 'data:image/png;base64,' + blobInfo.base64()

// success(img)

// this.$emit('imgUpload', blobInfo, success, failure)

if(blobInfo.size / 1024 / 1024 < 8){

failure('文件体积过大')

}

else if(!this.type.includes(blobInfo.blob().type)) {

failure('文件格式无效')

console.log('文件格式无效')

}

else{

console.log('上传')

let formData = new FormData()

formData.append("file",blobInfo.blob())

axios.post(this.$store.state.settings.uploadUrl,formData)

.then((res) => {

console.log(res)

if (res.code === 200) {

console.log(this.$store.state.settings.callbackUrl+res.data)

success(this.$store.state.settings.callbackUrl+res.data)

} else {

failure('上传失败!')

}

})

}

}

},

}

},

created() {

this.\_isEdit = this.isEdit

if (this.isEdit) {

const id = this.$route.params && this.$route.params.id

this.fetchData(id)

}

// Why need to make a copy of this.$route here?

// Because if you enter this page and quickly switch tag, may be in the execution of the setTagsViewTitle function, this.$route is no longer pointing to the current page

// https://github.com/PanJiaChen/vue-element-admin/issues/1221

this.tempRoute = Object.assign({}, this.$route)

},

mounted() {

tinymce.init({

})

},

methods: {

fetchData(id) {

WebArticleApi.findOne(id).then(data => {

this.postForm = data

this.setPageTitle()

}).catch(err => {

console.log(err)

})

},

setPageTitle() {

const title = '编辑文章'

document.title = `${title} : ${this.postForm.id}`

},

submitForm(value) {

this.$refs.postForm.validate((valid) => {

if (valid) {

this.loading = true

console.log(this.isEdit)

if(this.\_isEdit){

WebArticleApi.update(this.postForm)

}

else{

WebArticleApi.add(this.postForm).then(data => {

console.log(data)

this.postForm.id = data

this.setPageTitle()

}).catch(err => {

console.log(err)

})

this.\_isEdit = true

}

if(value === 1){

this.$notify({

title: '成功',

message: '发布文章成功',

type: 'success',

duration: 2000

})

this.postForm.status = 'reviewed'

this.loading = false

if(this.isEdit){

this.$router.push({path:'../list'})

}

else{

this.$router.push({path:'./list'})

}

}

else if(value === 3) {

this.$notify({

title: '成功',

message: '更新成功',

type: 'success',

duration: 2000

})

this.postForm.status = 'reviewing'

this.loading = false

if(this.isEdit){

this.$router.push({path:'../list'})

}

else{

this.$router.push({path:'./list'})

}

}

else {

this.$notify({

title: '成功',

message: '更新成功',

type: 'success',

duration: 2000

})

this.postForm.status = 'reject'

this.loading = false

if(this.isEdit){

this.$router.push({path:'../list'})

}

else{

this.$router.push({path:'./list'})

}

}

} else {

console.log('error submit!!')

return false

}

})

},

// 显示图片上传模块

imageCropperShow() {

this.cropperShow = !this.cropperShow

},

// 图片上传成功后执行

cropUploadSuccess(jsonData, field){

this.postForm.pic = jsonData.data

const tempData = Object.assign({}, this.postForm)

WebArticleApi.update(tempData).then(() => {

this.$notify({

title: 'success',

message: '图片上传成功',

type: 'success',

duration: 2000

})

this.getList()

})

},

picShow(pic) {

this.picVisible = !this.picVisible

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + pic

},

// 输入事件

handleInput(value) {

// console.log(value)

},

// 点击事件

handleClick(e, editor) {

// console.log(e, editor)

},

// 清空事件

clearClick() {

this.postForm.article = ''

},

// 禁用事件

disableClick() {

this.isEditorDisabled = !this.isEditorDisabled

}

}

}

</script>

<style lang="scss" scoped>

@import "~@/styles/mixin.scss";

.createPost-container {

position: relative;

.createPost-main-container {

padding: 40px 45px 20px 50px;

.postInfo-container {

position: relative;

@include clearfix;

margin-bottom: 10px;

.postInfo-container-item {

float: left;

}

}

}

.word-counter {

width: 40px;

position: absolute;

right: 10px;

top: 0px;

}

}

.article-textarea /deep/ {

textarea {

padding-right: 40px;

resize: none;

border: none;

border-radius: 0px;

border-bottom: 1px solid #bfcbd9;

}

}

</style>

<template>

<el-dropdown :show-timeout="100" trigger="click">

<el-button plain>

{{ !comment\_disabled?'Comment: opened':'Comment: closed' }}

<i class="el-icon-caret-bottom el-icon--right" />

</el-button>

<el-dropdown-menu slot="dropdown" class="no-padding">

<el-dropdown-item>

<el-radio-group v-model="comment\_disabled" style="padding: 10px;">

<el-radio :label="true">

Close comment

</el-radio>

<el-radio :label="false">

Open comment

</el-radio>

</el-radio-group>

</el-dropdown-item>

</el-dropdown-menu>

</el-dropdown>

</template>

<script>

export default {

props: {

value: {

type: Boolean,

default: false

}

},

computed: {

comment\_disabled: {

get() {

return this.value

},

set(val) {

this.$emit('input', val)

}

}

}

}

</script>

<template>

<article-detail :is-edit="false" />

</template>

<script>

import ArticleDetail from './components/ArticleDetail'

export default {

name: 'CreateArticle',

components: { ArticleDetail }

}

</script>

<template>

<article-detail :is-edit="true" />

</template>

<script>

import ArticleDetail from './components/ArticleDetail'

export default {

name: 'EditForm',

components: { ArticleDetail }

}

</script>

<template>

</template>

<script>

export default {

name: "index"

}

</script>

<style scoped>

</style>

<template>

<div>

<router-view />

</div>

</template>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleCreate">

新增

</el-button>

<el-select v-model="filterIsDel" style="width: 140px" class="filter-item" @change="handleFilter">

<el-option v-for="item in filterMap" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</div>

<el-table

:key="tableKey"

v-loading="listLoading"

:data="list"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.createAt }}</span>

</template>

</el-table-column>

<el-table-column label="视频标题" min-width="120px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoTitle }}</span>

</template>

</el-table-column>

<el-table-column label="创作团队" width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoAuthor }}</span>

</template>

</el-table-column>

<el-table-column label="视频链接" width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoUrl }}</span>

</template>

</el-table-column>

<el-table-column label="视频简介" max-width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoProfile }}</span>

</template>

</el-table-column>

<el-table-column label="分发" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-button v-if="row.videoStatus==='reviewed'" size="mini" type="success" @click="handleModifyStatus(row,'processing')">

分发

</el-button>

<el-tag v-if="row.videoStatus!=='reviewed' && row.videoStatus!=='processing' && row.videoStatus!=='complete'">

请先通过审核

</el-tag>

<el-tag v-if="row.videoStatus==='complete'" :type="row.videoStatus | statusFilter">

已分发

</el-tag>

<el-tag v-if="row.videoStatus==='processing'" :type="row.videoStatus | statusFilter">

分发中

</el-tag>

</template>

</el-table-column>

<el-table-column label="状态" class-name="status-col" width="100px">

<template slot-scope="{row}">

<el-tag :type="row.videoStatus | statusFilter">

{{ row.videoStatus | statusNameFilter}}

</el-tag>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-button v-show="row.videoStatus==='reviewing'" type="primary" size="mini" @click="handleUpdate(row)">

审阅

</el-button>

<el-button v-show="row.videoStatus==='reviewed'" size="mini" type="success" @click="handleUpdate(row)">

修改

</el-button>

<el-button v-show="row.videoStatus==='reject'" size="mini" @click="handleModifyStatus(row,'reviewing')">

待定

</el-button>

<el-button v-show="row.videoStatus==='reviewed'" size="mini" type="danger" @click="handleModifyStatus(row,'reject')">

驳回

</el-button>

<el-tag v-show="row.videoStatus==='processing'" :type="row.videoStatus | statusFilter">

分发中

</el-tag>

<el-tag v-show="row.videoStatus==='complete'" :type="row.videoStatus | statusFilter">

已分发

</el-tag>

</template>

</el-table-column>

</el-table>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<el-dialog :title="textMap" :visible.sync="dialogFormVisible">

<el-form ref="dataForm" :rules="rules" :model="temp" label-position="left" label-width="100px" style="width: 400px; margin-left:50px;">

<el-form-item label="审核人" prop="videoReviewer">

<el-input v-model="temp.videoReviewer" />

</el-form-item>

<el-form-item label="视频标题" prop="videoTitle">

<el-input v-model="temp.videoTitle" />

</el-form-item>

<el-form-item label="创作团队" prop="videoAuthor">

<el-input v-model="temp.videoAuthor" />

</el-form-item>

<el-form-item label="视频简介" prop="videoProfile">

<el-input v-model="temp.videoProfile" :autosize="{ minRows: 2, maxRows: 4}" type="textarea" placeholder="请输入简介" />

</el-form-item>

<el-form-item label="视频封面" prop="videoPic">

<el-button v-show="!temp.videoPic" type="primary" size="mini" @click="imageCropperShow()">

上传图片

</el-button>

<el-button v-show="temp.videoPic" type="primary" size="mini" @click="imageCropperShow()">

更新图片

</el-button>

<el-button v-show="temp.videoPic" type="primary" size="mini" @click="picShow(temp.videoPic)">

预览

</el-button>

</el-form-item>

<el-form-item label="上传视频" prop="videoPath">

<el-upload

:action="this.$store.state.settings.uploadUrl"

:show-file-list="false"

list-type="picture-card"

:on-success="answerVideoSuccess"

:before-upload="beforeUpload"

:on-progress="uploadVideoProcess"

:on-remove="handleRemove"

:on-error="handleError"

accept="video/mp4,video/flv,video/avi,video/wmv"

style="margin-top: 5%;">

<video v-if="answerVideoUrl && videoFlag === false" controls="controls" :src="answerVideoUrl" class="avatar"></video>

<i v-if="!answerVideoUrl && videoFlag === false" class="el-icon-plus avatar-uploader-icon"></i>

<el-progress v-if="videoFlag === true" type="circle" :percentage="percent" style="margin-top: 10px"></el-progress>

</el-upload>

</el-form-item>

<el-form-item label="视频分区" prop="videoClass">

<el-select v-model="temp.videoClass" class="filter-item" placeholder="类目一览" value="">

<el-option v-for="item in videoClassOptions" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</el-form-item>

<el-form-item label="视频标签" prop="videoTag">

<el-tag

:key="tag"

v-for="tag in temp.videoTag"

closable

:disable-transitions="false"

@close="handleClose(tag)">

{{tag}}

</el-tag>

<el-input

class="input-new-tag"

v-if="inputVisible"

v-model="inputValue"

ref="saveTagInput"

size="small"

@keyup.enter.native="handleInputConfirm"

@blur="handleInputConfirm"

>

</el-input>

<el-button v-else class="button-new-tag" size="small" @click="showInput">+ 新标签</el-button>

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer">

<el-button @click="handleCancel()">

取消

</el-button>

<el-button v-if="temp.videoStatus!=='reviewed'" type="primary" @click="dialogStatus==='create'?createData():updateData('reviewed')">

过审

</el-button>

<el-button v-if="dialogStatus!=='create' && temp.videoStatus!=='reviewed'" type="danger" @click="updateData('reject')">

驳回

</el-button>

<el-button v-if="temp.videoStatus==='reviewed'" type="primary" @click="updateData('reviewed')">

保存

</el-button>

</div>

</el-dialog>

<el-dialog title="提示" :visible.sync="delVisible" width="300px" center>

<div class="del-dialog-cnt">删除不可恢复，是否确定删除？</div>

<span slot="footer" class="dialog-footer">

<el-button @click="delVisible = false">取 消</el-button>

<el-button type="primary" @click="handleDelete(row)" >确 定</el-button>

</span>

</el-dialog>

<my-upload

method="POST"

field="file"

v-model="cropperShow"

:width=1200

:height=750

:url="this.$store.state.settings.uploadUrl"

lang-type='zh'

img-format='jpg'

img-bgc='#FFF'

:no-circle=true

@crop-upload-success="cropUploadSuccess">

</my-upload>

<el-dialog title="图片预览" :visible.sync="picVisible" width="600px" center>

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

<img :src="answerPicImageUrl" alt="" class="title-pic">

</div>

<span slot="footer" class="dialog-footer">

<el-button @click="picVisible = false">确定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import MediaVideoApi from "@/api/media/MediaVideoApi";

import WebVerifyApi from "@/api/admin/WebVerifyApi";

import 'babel-polyfill'; // es6 shim

import myUpload from 'vue-image-crop-upload';

export default {

name: 'VerifyMediaVideo',

components: { Pagination, myUpload },

directives: { waves },

filters: {

statusNameFilter(status) {

const statusMap = {

reviewing: '审核中',

draft: '草稿',

reviewed: '已审核',

reject: '驳回',

processing: '分发中',

complete: '分发完毕'

}

return statusMap

},

statusFilter(status) {

const statusMap = {

reviewing: 'warning',

reviewed: 'success',

draft: 'info',

reject: 'danger',

processing: 'warning',

}

return statusMap

}

},

data() {

return {

videoFlag: false,

percent: 0,

inputVisible: false,

inputValue: '',

cropperShow: false,

picVisible: false,

filterIsDel: 1,

delVisible: false,

tableKey: 0,

list: null,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC',

active: ''

},

// 已过审的不允许删除和存为草稿和提交审核

statusOptions: [{

label: '提交审核',

value: 'reviewing'

}, {

label: '存为草稿',

value: 'draft'

}],

videoClassOptions: [{

label: '日常',

value: '21'

}, {

label: '搞笑',

value: '138'

}, {

label: '美食',

value: '76'

}, {

label: '科普',

value: '124'

}, {

label: '技术',

value: '122'

}, {

label: '动物',

value: '75'

}, {

label: '运动',

value: '163'

},

],

showReviewer: false,

temp: {

videoAuthor: '',

videoProfile: '',

videoTitle: '',

videoStatus: '',

videoUrl: '',

videoReviewer: '',

videoPic: '',

videoClass: '',

videoTag: ,

videoPath: '',

updateAt:'',

createAt:'',

IsDel: ''

},

videoTagTemp: undefined,

dialogFormVisible: false,

dialogStatus: '',

textMap: {

update: '编辑',

create: '创建'

},

filterMap: [

{ label: '全部', value: 1 },

{ label: '未分发', value: 'false' },

{ label: '已分发', value: 'true'}

],

rules: {

videoTitle: ,

videoReviewer: ,

videoProfile: ,

videoClass: ,

videoPic: ,

videoPath: ,

},

downloadLoading: false,

answerPicImageUrl: '',

answerVideoUrl: ''

}

},

created() {

this.getList()

},

methods: {

// 刷新界面

getList() {

this.listLoading = true

if(this.filterIsDel === 1){

WebVerifyApi.findAllMediaVideo(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

Object.keys(this.list).forEach(key => (this.list.videoTag = JSON.parse(this.list.videoTag)))

})

}

else{

this.listQuery.active = this.filterIsDel

WebVerifyApi.findClassMediaVideo(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

Object.keys(this.list).forEach(key => (this.list.videoTag = JSON.parse(this.list.videoTag)))

})

}

this.listLoading = false

},

handleFilter() {

this.listQuery.page = 1

this.getList()

},

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.filterIsDel = 1

this.handleFilter()

},

handleCreate() {

this.answerPicImageUrl = ''

this.answerVideoUrl = ''

Object.keys(this.temp).forEach(key => (this.temp = ''))

this.temp.videoTag =

this.dialogStatus = 'create'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

createData() {

this.$refs.validate((valid) => {

if (valid) {

this.temp.videoStatus = 'reviewed'

this.videoTagTemp = this.temp.videoTag

this.dialogFormVisible = false

this.temp.videoTag = JSON.stringify(this.temp.videoTag)

const tempData = Object.assign({}, this.temp)

this.temp.videoTag = this.videoTagTemp

MediaVideoApi.add(tempData).then(() => {

this.list.unshift(this.temp)

this.getList()

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

})

}

})

},

handleUpdate(row) {

this.temp = Object.assign({}, row) // copy obj

if(this.temp.videoPic){

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + this.temp.videoPic

}

if(this.temp.videoPath){

this.answerVideoUrl = this.$store.state.settings.callbackUrl + this.temp.videoPath

}

this.dialogStatus = 'update'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

handleModifyStatus(row, status) {

this.listLoading = true

row.videoStatus = status

this.temp = Object.assign({}, row) // copy obj

this.videoTagTemp = this.temp.videoTag

this.temp.videoTag = JSON.stringify(this.temp.videoTag)

const tempData = Object.assign({}, this.temp)

this.temp.videoTag = this.videoTagTemp

if(this.temp.videoStatus === 'processing'){

WebVerifyApi.mediaVideoDistribution(tempData).then(() => {})

}

else{

MediaVideoApi.update(tempData).then(() => {})

}

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

updateData(status) {

this.$refs.validate((valid) => {

if (valid) {

this.temp.videoStatus = status

this.videoTagTemp = this.temp.videoTag

this.dialogFormVisible = false

this.temp.videoTag = JSON.stringify(this.temp.videoTag)

const tempData = Object.assign({}, this.temp)

this.temp.videoTag = this.videoTagTemp

MediaVideoApi.update(tempData).then(() => {

this.getList()

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

})

}

})

},

handleCancel(){

this.dialogFormVisible = false

this.answerPicImageUrl = ''

this.answerVideoUrl = ''

},

confirmDelete(row) {

// this.delVisible = true

if(confirm('确定要删除吗')===true){

this.handleDelete(row)

}

},

handleDelete(row) {

MediaVideoApi.delOne(row.id).then(data => {

this.getList()

})

this.$notify({

title: 'Success',

message: '成功删除',

type: 'success',

duration: 2000

})

const index = this.list.indexOf(row)

this.list.splice(index, 1)

},

answerPicImageSuccess (res, file) {

this.temp.videoPic = res.data

this.answerPicImageUrl = URL.createObjectURL(file.raw)

},

beforeUpload (file) {

// 只允许上传2GB以内大小的图片

const isLt2G = file.size / 1024 / 1024 / 1024 < 2;

if (!isLt2G) {

this.$message.error('上传视频大小不能超过2GB!');

}

return isLt2G;

},

handleClose(tag) {

this.temp.videoTag.splice(this.temp.videoTag.indexOf(tag), 1);

},

showInput() {

this.inputVisible = true;

this.$nextTick(\_ => {

this.$refs.saveTagInput.$refs.input.focus();

});

},

handleInputConfirm() {

let inputValue = this.inputValue;

if (inputValue) {

console.log(inputValue)

this.temp.videoTag.push(inputValue);

}

this.inputVisible = false;

this.inputValue = '';

},

picShow(pic) {

this.picVisible = !this.picVisible

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + pic

},

// 显示图片上传模块

imageCropperShow() {

this.cropperShow = !this.cropperShow

},

// 图片上传成功后执行

cropUploadSuccess(jsonData, field){

this.temp.videoPic = jsonData.data

this.$notify({

title: 'success',

message: '图片上传成功',

type: 'success',

duration: 2000

})

},

uploadVideoProcess(event, file, fileList){

this.videoFlag = true;

this.percent = Math.floor(event.percent)

},

handleRemove(file, fileList) {

this.temp.videoPath = '';

},

answerVideoSuccess(res, file){

this.temp.videoPath = res.data

this.answerVideoUrl = URL.createObjectURL(file.raw)

this.videoFlag = false;

},

handleError(err, file, fileList) {

// 上传失败异常处理

const error = JSON.parse(JSON.stringify(err));

console.log(err)

console.log(error)

this.$message.error(error.status.toString());

this.videoFlag = false;

this.percent = 0;

},

}

}

</script>

<style>

.el-tag + .el-tag {

margin-left: 10px;

}

.button-new-tag {

margin-left: 10px;

height: 32px;

line-height: 30px;

padding-top: 0;

padding-bottom: 0;

}

.input-new-tag {

width: 90px;

margin-left: 10px;

vertical-align: bottom;

}

.title-pic{

width: 90%;

}

.avatar-uploader .el-upload {

border: 1px dashed #d9d9d9;

border-radius: 6px;

cursor: pointer;

position: relative;

overflow: hidden;

width: 178px;

height: 178px;

}

.avatar-uploader .el-upload:hover {

border-color: #409EFF;

}

.avatar-uploader-icon {

line-height: 140px;

}

.avatar {

height: 144px;

}

.image-preview {

width: 178px;

height: 178px;

position: relative;

border: 1px dashed #d9d9d9;

border-radius: 6px;

float: left;

}

.image-preview .image-preview-wrapper {

position: relative;

width: 100%;

height: 100%;

}

.image-preview .image-preview-wrapper img {

width: 100%;

height: 100%;

}

.image-preview .image-preview-action {

position: absolute;

width: 100%;

height: 100%;

left: 0;

top: 0;

cursor: default;

text-align: center;

color: #fff;

opacity: 0;

font-size: 20px;

background-color: rgba(0, 0, 0, .5);

transition: opacity .3s;

cursor: pointer;

text-align: center;

line-height: 200px;

}

.image-preview .image-preview-action .el-icon-delete {

font-size: 32px;

}

.image-preview:hover .image-preview-action {

opacity: 1;

}

.el-upload--picture-card{

display: block;

width: 258px;

height: 146px;

overflow: hidden;

}

</style>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleCreate">

新增

</el-button>

<el-select v-model="filterIsDel" style="width: 140px" class="filter-item" @change="handleFilter">

<el-option v-for="item in filterMap" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</div>

<el-table

:key="tableKey"

v-loading="listLoading"

:data="list"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

>

<el-table-column label="创建日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.createAt }}</span>

</template>

</el-table-column>

<el-table-column label="员工姓名" min-width="120px" align="center">

<template slot-scope="{row}">

<span>{{ row.name }}</span>

</template>

</el-table-column>

<el-table-column label="员工昵称" width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.nickname }}</span>

</template>

</el-table-column>

<el-table-column label="员工等级" width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.level }}</span>

</template>

</el-table-column>

<el-table-column label="状态" class-name="status-col" width="100px">

<template slot-scope="{row}">

<el-tag :type="row.isDel | statusFilter">

{{ row.isDel | statusNameFilter}}

</el-tag>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-button v-if="row.isDel === true" size="mini" @click="handleModifyStatus(row, 'false')">

撤回

</el-button>

<el-button v-if="row.isDel === false" size="mini" type="danger" @click="handleModifyStatus(row, 'true')">

删除

</el-button>

</template>

</el-table-column>

</el-table>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<el-dialog :title="textMap" :visible.sync="dialogFormVisible">

<el-form ref="dataForm" :rules="rules" :model="temp" label-position="left" label-width="100px" style="width: 400px; margin-left:50px;">

<el-form-item label="员工姓名" prop="name">

<el-input v-model="temp.name" />

</el-form-item>

<el-form-item label="员工昵称" prop="nickname">

<el-input v-model="temp.nickname" />

</el-form-item>

<el-form-item label="员工等级" prop="level">

<el-select v-model="temp.level" class="filter-item" placeholder="管理员/普通员工">

<el-option v-for="item in statusOptions" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer">

<el-button @click="dialogFormVisible = false">

取消

</el-button>

<el-button type="primary" @click="dialogStatus==='create'?createData():updateData()">

确认

</el-button>

</div>

</el-dialog>

<el-dialog title="提示" :visible.sync="delVisible" width="300px" center>

<div class="del-dialog-cnt">删除不可恢复，是否确定删除？</div>

<span slot="footer" class="dialog-footer">

<el-button @click="delVisible = false">取 消</el-button>

<el-button type="primary" @click="handleDelete(row)" >确 定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import WebAccountApi from "@/api/admin/WebAccountApi";

export default {

name: 'bbb',

components: { Pagination },

directives: { waves },

filters: {

statusNameFilter(status) {

// this.status.toString()

const statusMap = {

true: '已删除',

false: '正常',

}

return statusMap

},

statusFilter(status) {

// this.status.toString()

const statusMap = {

true: 'warning',

false: 'success',

}

return statusMap

}

},

data() {

return {

filterIsDel: 2,

delVisible: false,

tableKey: 0,

list: null,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC',

active: 'true'

},

statusOptions: [{

label: '管理员',

value: '1'

}, {

label: '普通员工',

value: '2'

}],

showReviewer: false,

temp: {

name: '',

nickname: '',

level: '',

createAt:'',

isDel: ''

},

dialogFormVisible: false,

dialogStatus: '',

textMap: {

update: '编辑',

create: '创建'

},

filterMap: [{

label: '正常',

value: 2

}, {

label: '已删除',

value: 3

}],

rules: {

name: ,

nickname: ,

},

downloadLoading: false,

answerPicImageUrl: ''

}

},

created() {

this.getList()

},

methods: {

// 刷新界面

getList() {

this.listLoading = true

if(this.filterIsDel === 3){

this.listQuery.active = 'true'

WebAccountApi.findAllByKeywords(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

}

else{

this.listQuery.active = 'false'

WebAccountApi.findAllByKeywords(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

}

this.listLoading = false

},

handleFilter() {

this.listQuery.page = 1

this.getList()

},

handleModifyStatus(row, status) {

this.listLoading = true

WebAccountApi.mod(row.id, status).then(() => {

this.getList()

})

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.filterIsDel = 1

this.handleFilter()

},

handleCreate() {

Object.keys(this.temp).forEach(key => (this.temp = ''))

this.dialogStatus = 'create'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

createData() {

this.$refs.validate((valid) => {

if (valid) {

WebAccountApi.add(this.temp).then(() => {

this.list.unshift(this.temp)

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

})

}

})

},

handleUpdate(row) {

this.temp = Object.assign({}, row) // copy obj

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + this.temp.videoPic

this.dialogStatus = 'update'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

updateData() {

this.$refs.validate((valid) => {

if (valid) {

const tempData = Object.assign({}, this.temp)

WebAccountApi.update(tempData).then(() => {

for (const v of this.list) {

if (v.id === this.temp.id) {

const index = this.list.indexOf(v)

this.list.splice(index, 1, this.temp)

break

}

}

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

})

}

})

},

handleDelete(row) {

WebAccountApi.delOne(row.id).then(data => {

this.getList()

})

this.$notify({

title: 'Success',

message: '成功删除',

type: 'success',

duration: 2000

})

const index = this.list.indexOf(row)

this.list.splice(index, 1)

},

answerPicImageSuccess (res, file) {

this.temp.videoPic = res.data

this.answerPicImageUrl = URL.createObjectURL(file.raw)

},

beforeUpload (file) {

// 只允许上传8M以内大小的图片

const isLt8M = file.size / 1024 / 1024 < 8;

if (!isLt8M) {

this.$message.error('上传头像图片大小不能超过8MB!');

}

return isLt8M;

},

}

}

</script>

<template>

<div>

<router-view />

</div>

</template>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleCreate">

新增

</el-button>

<el-select v-model="filterIsDel" style="width: 140px" class="filter-item" @change="handleFilter">

<el-option v-for="item in filterMap" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</div>

<el-table

:key="tableKey"

v-loading="listLoading"

:data="list"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.updateAt }}</span>

</template>

</el-table-column>

<el-table-column label="轮播图标题" min-width="120px" align="center">

<template slot-scope="{row}">

<span v-if="row.swiperStatus==='reviewed' || row.swiperStatus==='reviewing'">{{ row.swiperName }}</span>

<span v-else class="link-type" @click="handleUpdate(row)">{{ row.swiperName }}</span>

</template>

</el-table-column>

<el-table-column label="上架/下架" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-button v-if="row.isDel===true" size="mini" type="success" @click="handleModifyIsDel(row, false)">

上架

</el-button>

<el-button v-if="row.isDel===false" size="mini" @click="handleModifyIsDel(row, true)">

下架

</el-button>

</template>

</el-table-column>

<el-table-column label="状态" class-name="status-col" width="100px">

<template slot-scope="{row}">

<el-tag :type="row.swiperStatus | statusFilter">

{{ row.swiperStatus | statusNameFilter}}

</el-tag>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-button v-if="row.swiperStatus==='reviewing'" type="primary" size="mini" @click="handleUpdate(row)">

审阅

</el-button>

<el-button v-if="row.swiperStatus==='reviewed'" size="mini" type="success" @click="handleUpdate(row)">

修改

</el-button>

<el-button v-if="row.swiperStatus==='reject'" size="mini" @click="handleModifyStatus(row,'reviewing')">

待定

</el-button>

<el-button v-if="row.swiperStatus==='reviewed'" size="mini" type="danger" @click="handleModifyStatus(row,'reject')">

驳回

</el-button>

</template>

</el-table-column>

</el-table>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<el-dialog :title="textMap" :visible.sync="dialogFormVisible">

<el-form ref="dataForm" :rules="rules" :model="temp" label-position="left" label-width="100px" style="width: 400px; margin-left:50px;">

<el-form-item label="轮播图标题" prop="swiperName">

<el-input v-model="temp.swiperName" />

</el-form-item>

<!-- <el-form-item label="轮播图">-->

<!-- <el-upload-->

<!-- :action="this.$store.state.settings.uploadUrl"-->

<!-- :show-file-list="false"-->

<!-- :on-success="answerPicImageSuccess"-->

<!-- :before-upload="beforeUpload"-->

<!-- drag-->

<!-- prop="pic"-->

<!-- accept="image/png,image/gif,image/jpg,image/jpeg"-->

<!-- class="upload-demo" style="margin-top: 5%">-->

<!-- <img v-if="answerPicImageUrl" :src="answerPicImageUrl">-->

<!-- <i class="el-icon-upload"></i>-->

<!-- <div class="el-upload\_\_text">将文件拖到此处，或<em>点击上传</em></div>-->

<!-- </el-upload>-->

<!-- </el-form-item>-->

<el-form-item label="轮播图" prop="swiperPic">

<el-button v-show="!temp.swiperPic" type="primary" size="mini" @click="imageCropperShow()">

上传图片

</el-button>

<el-button v-show="temp.swiperPic" type="primary" size="mini" @click="imageCropperShow()">

更新图片

</el-button>

<el-button v-show="temp.swiperPic" type="primary" size="mini" @click="picShow(temp.swiperPic)">

预览

</el-button>

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer">

<el-button @click="dialogFormVisible = false">

取消

</el-button>

<el-button v-if="temp.swiperStatus!=='reviewed'" type="primary" @click="dialogStatus==='create'?createData():updateData('reviewed')">

过审

</el-button>

<el-button v-if="dialogStatus!=='create' && temp.swiperStatus!=='reviewed'" type="danger" @click="updateData('reject')">

驳回

</el-button>

<el-button v-if="temp.swiperStatus==='reviewed'" type="primary" @click="updateData('reviewed')">

保存

</el-button>

</div>

</el-dialog>

<el-dialog title="提示" :visible.sync="delVisible" width="300px" center>

<div class="del-dialog-cnt">删除不可恢复，是否确定删除？</div>

<span slot="footer" class="dialog-footer">

<el-button @click="delVisible = false">取 消</el-button>

<el-button type="primary" @click="handleDelete(row)" >确 定</el-button>

</span>

</el-dialog>

<my-upload

method="POST"

field="file"

v-model="cropperShow"

:width=1920

:height=450

:url="this.$store.state.settings.uploadUrl"

lang-type='zh'

img-format='jpg'

img-bgc='#FFF'

:no-circle=true

@crop-upload-success="cropUploadSuccess">

</my-upload>

<el-dialog title="图片预览" :visible.sync="picVisible" width="800px" center>

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

<img :src="answerPicImageUrl" alt="" class="title-pic">

</div>

<span slot="footer" class="dialog-footer">

<el-button @click="picVisible = false">确定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import WebSwiperApi from "@/api/website/WebSwiperApi";

import WebVerifyApi from "@/api/admin/WebVerifyApi";

import 'babel-polyfill'; // es6 shim

import myUpload from 'vue-image-crop-upload';

export default {

name: 'swiper',

components: { Pagination, myUpload },

directives: { waves },

filters: {

statusNameFilter(status) {

const statusMap = {

reviewing: '审核中',

draft: '草稿',

reviewed: '已审核',

reject: '驳回'

}

return statusMap

},

statusFilter(status) {

const statusMap = {

reviewing: 'warning',

reviewed: 'success',

draft: 'info',

reject: 'danger'

}

return statusMap

}

},

data() {

return {

cropperShow: false,

picVisible: false,

filterIsDel: 1,

delVisible: false,

tableKey: 0,

list: null,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC',

active: ''

},

// 已过审的不允许删除和存为草稿和提交审核

statusOptions: [{

label: '提交审核',

value: 'reviewing'

}, {

label: '存为草稿',

value: 'draft'

}],

showReviewer: false,

temp: {

swiperName: "",

swiperStatus: '',

swiperPic: '',

swiperReviewer: '',

createAt:'',

IsDel: ''

},

dialogFormVisible: false,

dialogStatus: '',

textMap: {

update: '编辑',

create: '创建'

},

filterMap: [

{ label: '全部', value: 1 },

{ label: '未下架', value: 'false' },

{ label: '已下架', value: 'true'}

],

rules: {

swiperName: ,

swiperStatus: ,

swiperPic: ,

},

downloadLoading: false,

answerPicImageUrl: ''

}

},

created() {

this.getList()

},

methods: {

// 刷新界面

getList() {

this.listLoading = true

if(this.filterIsDel === 1){

WebVerifyApi.findAllSwiper(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

}

else{

this.listQuery.active = this.filterIsDel

WebVerifyApi.findClassSwiper(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

}

this.listLoading = false

},

handleFilter() {

this.listQuery.page = 1

this.getList()

},

handleModifyStatus(row, status) {

this.listLoading = true

row.swiperStatus = status

this.temp = Object.assign({}, row) // copy obj

WebSwiperApi.update(this.temp)

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

handleModifyIsDel(row, active) {

this.listLoading = true

row.isDel = active

this.temp = Object.assign({}, row) // copy obj

WebSwiperApi.update(this.temp)

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.filterIsDel = 1

this.handleFilter()

},

handleCreate() {

Object.keys(this.temp).forEach(key => (this.temp = ''))

this.dialogStatus = 'create'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

createData() {

this.$refs.validate((valid) => {

if (valid) {

WebSwiperApi.add(this.temp).then(() => {

this.list.unshift(this.temp)

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

})

}

})

},

handleUpdate(row) {

this.temp = Object.assign({}, row) // copy obj

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + this.temp.swiperPic

this.dialogStatus = 'update'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

updateData(status) {

this.$refs.validate((valid) => {

if (valid) {

this.temp.swiperStatus = status

const tempData = Object.assign({}, this.temp)

WebSwiperApi.update(tempData).then(() => {

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

})

}

})

},

confirmDelete(row) {

// this.delVisible = true

if(confirm('确定要删除吗')===true){

this.handleDelete(row)

}

},

handleDelete(row) {

WebSwiperApi.delOne(row.id).then(data => {

this.getList()

})

this.$notify({

title: 'Success',

message: '成功删除',

type: 'success',

duration: 2000

})

const index = this.list.indexOf(row)

this.list.splice(index, 1)

},

answerPicImageSuccess (res, file) {

this.temp.swiperPic = res.data

this.answerPicImageUrl = URL.createObjectURL(file.raw)

},

beforeUpload (file) {

// 只允许上传8M以内大小的图片

const isLt8M = file.size / 1024 / 1024 < 8;

if (!isLt8M) {

this.$message.error('上传头像图片大小不能超过8MB!');

}

return isLt8M;

},

picShow(pic) {

this.picVisible = !this.picVisible

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + pic

},

// 显示图片上传模块

imageCropperShow() {

this.cropperShow = !this.cropperShow

},

// 图片上传成功后执行

cropUploadSuccess(jsonData, field){

this.temp.swiperPic = jsonData.data

this.$notify({

title: 'success',

message: '图片上传成功',

type: 'success',

duration: 2000

})

},

}

}

</script>

<style>

.title-pic{

width: 90%;

}

</style>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleCreate">

新增

</el-button>

<el-select v-model="filterIsDel" style="width: 140px" class="filter-item" @change="handleFilter">

<el-option v-for="item in filterMap" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</div>

<el-table

:key="tableKey"

v-loading="listLoading"

:data="list"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.createAt }}</span>

</template>

</el-table-column>

<el-table-column label="视频标题" min-width="120px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoTitle }}</span>

</template>

</el-table-column>

<el-table-column label="创作团队" width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoAuthor }}</span>

</template>

</el-table-column>

<el-table-column label="视频链接" width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoUrl }}</span>

</template>

</el-table-column>

<el-table-column label="视频简介" max-width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoProfile }}</span>

</template>

</el-table-column>

<el-table-column label="上架/下架" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-button v-if="row.isDel===true" size="mini" type="success" @click="handleModifyIsDel(row, false)">

上架

</el-button>

<el-button v-if="row.isDel===false" size="mini" @click="handleModifyIsDel(row, true)">

下架

</el-button>

</template>

</el-table-column>

<el-table-column label="状态" class-name="status-col" width="100px">

<template slot-scope="{row}">

<el-tag :type="row.videoStatus | statusFilter">

{{ row.videoStatus | statusNameFilter}}

</el-tag>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-button v-if="row.videoStatus==='reviewing'" type="primary" size="mini" @click="handleUpdate(row)">

审阅

</el-button>

<el-button v-if="row.videoStatus==='reviewed'" size="mini" type="success" @click="handleUpdate(row)">

修改

</el-button>

<el-button v-if="row.videoStatus==='reject'" size="mini" @click="handleModifyStatus(row,'reviewing')">

待定

</el-button>

<el-button v-if="row.videoStatus==='reviewed'" size="mini" type="danger" @click="handleModifyStatus(row,'reject')">

驳回

</el-button>

</template>

</el-table-column>

</el-table>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<el-dialog :title="textMap" :visible.sync="dialogFormVisible">

<el-form ref="dataForm" :rules="rules" :model="temp" label-position="left" label-width="100px" style="width: 400px; margin-left:50px;">

<el-form-item label="审核人" prop="videoReviewer">

<el-input v-model="temp.videoReviewer" />

</el-form-item>

<el-form-item label="视频标题" prop="videoTitle">

<el-input v-model="temp.videoTitle" />

</el-form-item>

<el-form-item label="创作团队" prop="author">

<el-input v-model="temp.videoAuthor" />

</el-form-item>

<el-form-item label="视频简介" prop="profile">

<el-input v-model="temp.videoProfile" :autosize="{ minRows: 2, maxRows: 4}" type="textarea" placeholder="请输入简介" />

</el-form-item>

<el-form-item label="视频链接" prop="url">

<el-input v-model="temp.videoUrl" />

</el-form-item>

<el-form-item label="视频封面" prop="videoPic">

<el-button v-show="!temp.videoPic" type="primary" size="mini" @click="imageCropperShow()">

上传图片

</el-button>

<el-button v-show="temp.videoPic" type="primary" size="mini" @click="imageCropperShow()">

更新图片

</el-button>

<el-button v-show="temp.videoPic" type="primary" size="mini" @click="picShow(temp.videoPic)">

预览

</el-button>

</el-form-item>

<!-- <el-form-item label="视频封面">-->

<!-- <el-upload-->

<!-- :action="this.$store.state.settings.uploadUrl"-->

<!-- :show-file-list="false"-->

<!-- :on-success="answerPicImageSuccess"-->

<!-- :before-upload="beforeUpload"-->

<!-- drag-->

<!-- prop="pic"-->

<!-- accept="image/png,image/gif,image/jpg,image/jpeg"-->

<!-- class="upload-demo" style="margin-top: 5%">-->

<!-- <img v-if="answerPicImageUrl" :src="answerPicImageUrl" class="video-pic-image">-->

<!-- <i class="el-icon-upload"></i>-->

<!-- <div class="el-upload\_\_text">将文件拖到此处，或<em>点击上传</em></div>-->

<!-- </el-upload>-->

<!-- </el-form-item>-->

</el-form>

<div slot="footer" class="dialog-footer">

<el-button @click="dialogFormVisible = false">

取消

</el-button>

<el-button v-if="temp.videoStatus!=='reviewed'" type="primary" @click="dialogStatus==='create'?createData():updateData('reviewed')">

过审

</el-button>

<el-button v-if="dialogStatus!=='create' && temp.videoStatus!=='reviewed'" type="danger" @click="updateData('reject')">

驳回

</el-button>

<el-button v-if="temp.videoStatus==='reviewed'" type="primary" @click="updateData('reviewed')">

保存

</el-button>

</div>

</el-dialog>

<el-dialog title="提示" :visible.sync="delVisible" width="300px" center>

<div class="del-dialog-cnt">删除不可恢复，是否确定删除？</div>

<span slot="footer" class="dialog-footer">

<el-button @click="delVisible = false">取 消</el-button>

<el-button type="primary" @click="handleDelete(row)" >确 定</el-button>

</span>

</el-dialog>

<my-upload

method="POST"

field="file"

v-model="cropperShow"

:width=600

:height=800

:url="this.$store.state.settings.uploadUrl"

lang-type='zh'

img-format='jpg'

img-bgc='#FFF'

:no-circle=true

@crop-upload-success="cropUploadSuccess">

</my-upload>

<el-dialog title="图片预览" :visible.sync="picVisible" width="400px" center>

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

<img :src="answerPicImageUrl" alt="" class="title-pic">

</div>

<span slot="footer" class="dialog-footer">

<el-button @click="picVisible = false">确定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import WebVideoApi from "@/api/website/WebVideoApi";

import WebVerifyApi from "@/api/admin/WebVerifyApi";

import 'babel-polyfill'; // es6 shim

import myUpload from 'vue-image-crop-upload';

export default {

name: 'VerifyVideo',

components: { Pagination, myUpload },

directives: { waves },

filters: {

statusNameFilter(status) {

const statusMap = {

reviewing: '审核中',

draft: '草稿',

reviewed: '已审核',

reject: '驳回'

}

return statusMap

},

statusFilter(status) {

const statusMap = {

reviewing: 'warning',

reviewed: 'success',

draft: 'info',

reject: 'danger'

}

return statusMap

}

},

data() {

return {

picVisible: false,

cropperShow: false,

filterIsDel: 1,

delVisible: false,

tableKey: 0,

list: null,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC',

active: ''

},

// 已过审的不允许删除和存为草稿和提交审核

statusOptions: [{

label: '提交审核',

value: 'reviewing'

}, {

label: '存为草稿',

value: 'draft'

}],

showReviewer: false,

temp: {

videoAuthor: "",

videoProfile: '',

videoTitle: '',

videoStatus: '',

videoUrl: '',

videoReviewer: '',

videoPic: '',

createAt:'',

IsDel: ''

},

dialogFormVisible: false,

dialogStatus: '',

textMap: {

update: '编辑',

create: '创建'

},

filterMap: [

{ label: '全部', value: 1 },

{ label: '未下架', value: 'false' },

{ label: '已下架', value: 'true'}

],

rules: {

videoTitle: ,

videoReviewer: ,

},

downloadLoading: false,

answerPicImageUrl: ''

}

},

created() {

this.getList()

},

methods: {

// 刷新界面

getList() {

this.listLoading = true

if(this.filterIsDel === 1){

WebVerifyApi.findAllVideo(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

}

else{

this.listQuery.active = this.filterIsDel

WebVerifyApi.findClassVideo(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

}

this.listLoading = false

},

handleFilter() {

this.listQuery.page = 1

this.getList()

},

handleModifyStatus(row, status) {

this.listLoading = true

row.videoStatus = status

this.temp = Object.assign({}, row) // copy obj

WebVideoApi.update(this.temp)

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

handleModifyIsDel(row, active) {

this.listLoading = true

row.isDel = active

this.temp = Object.assign({}, row) // copy obj

WebVideoApi.update(this.temp)

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.filterIsDel = 1

this.handleFilter()

},

handleCreate() {

Object.keys(this.temp).forEach(key => (this.temp = ''))

this.dialogStatus = 'create'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

createData() {

this.$refs.validate((valid) => {

if (valid) {

this.temp.videoStatus = 'reviewed'

WebVideoApi.add(this.temp).then(() => {

this.list.unshift(this.temp)

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

})

}

})

},

handleUpdate(row) {

this.temp = Object.assign({}, row) // copy obj

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + this.temp.videoPic

this.dialogStatus = 'update'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

updateData(status) {

this.$refs.validate((valid) => {

if (valid) {

this.temp.videoStatus = status

const tempData = Object.assign({}, this.temp)

WebVideoApi.update(tempData).then(() => {

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

})

}

})

},

confirmDelete(row) {

// this.delVisible = true

if(confirm('确定要删除吗')===true){

this.handleDelete(row)

}

},

handleDelete(row) {

WebVideoApi.delOne(row.id).then(data => {

this.getList()

})

this.$notify({

title: 'Success',

message: '成功删除',

type: 'success',

duration: 2000

})

const index = this.list.indexOf(row)

this.list.splice(index, 1)

},

answerPicImageSuccess (res, file) {

this.temp.videoPic = res.data

this.answerPicImageUrl = URL.createObjectURL(file.raw)

},

beforeUpload (file) {

// 只允许上传8M以内大小的图片

const isLt8M = file.size / 1024 / 1024 < 8;

if (!isLt8M) {

this.$message.error('上传头像图片大小不能超过8MB!');

}

return isLt8M;

},

picShow(pic) {

this.picVisible = !this.picVisible

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + pic

},

// 显示图片上传模块

imageCropperShow() {

this.cropperShow = !this.cropperShow

},

// 图片上传成功后执行

cropUploadSuccess(jsonData, field){

this.temp.videoPic = jsonData.data

this.$notify({

title: 'success',

message: '图片上传成功',

type: 'success',

duration: 2000

})

},

}

}

</script>

<style>

.title-pic{

width: 60%;

}

</style>

<template>

<div class="dashboard-container">

<el-row :gutter="60">

<el-col :span="6">

<div class="grid-content bg-purple">

<div class="data-name">

文章统计

</div>

<div class="text-content">

{{count.articleWebCount}}

</div>

</div>

</el-col>

<el-col :span="6">

<div class="grid-content bg-purple">

<div class="data-name">

视频统计

</div>

<div class="text-content">

{{count.videoWebCount}}

</div>

</div>

</el-col>

<el-col :span="6">

<div class="grid-content bg-purple">

<div class="data-name">

轮播图统计

</div>

<div class="text-content">

{{count.swiperWebCount}}

</div>

</div>

</el-col>

<el-col :span="6">

<div class="grid-content bg-purple">

<div class="data-name">

网站浏览量

</div>

<div class="text-content">

{{count.viewWebCount}}

</div>

</div>

</el-col>

</el-row>

<el-row :gutter="60">

<el-col :span="6">

<div class="grid-content bg-purple">

<div class="data-name">

活动统计

</div>

<div class="text-content">

{{count.activityMiniCount}}

</div>

</div>

</el-col>

<el-col :span="6">

<div class="grid-content bg-purple">

<div class="data-name">

赞助商统计

</div>

<div class="text-content">

{{count.videoWebCount}}

</div>

</div>

</el-col>

<el-col :span="6">

<div class="grid-content bg-purple">

<div class="data-name">

员工统计

</div>

<div class="text-content">

{{count.staffWebCount}}

</div>

</div>

</el-col>

<el-col :span="6">

<div class="grid-content bg-purple">

<div class="data-name">

小程序使用量

</div>

<div class="text-content">

{{count.userMiniCount}}

</div>

</div>

</el-col>

</el-row>

</div>

</template>

<script>

import LoginApi from '@/api/website/LoginApi'

export default {

data () {

return {

listLoading: true,

count:{

viewWebCount: null,

articleWebCount: null,

swiperWebCount: null,

userMiniCount: null,

videoWebCount: null,

merchantMiniCount: null,

staffWebCount: null,

activityMiniCount: null,

}

}

},

created() {

this.getList()

},

methods: {

getList() {

this.listLoading = true

LoginApi.getCount().then(data => {

this.count = data;

console.log(data)

})

this.listLoading = false

},

},

name: 'Dashboard',

computed: {

}

}

</script>

<style>

.el-row {

margin-bottom: 30px;

}

.dashboard-container{

margin: 40px 60px 30px 60px;

}

.bg-purple {

background: rgba(0,161,214,.03);

vertical-align: middle;

padding: 40px 0 0 0;

}

.grid-content {

border-radius: 4px;

min-height: 150px;

}

.data-name{

font-size: 18px;

line-height: 18px;

color: #757575;

text-align: center;

}

.section-item{

vertical-align: middle;

padding: 16px 18%;

}

.text-content{

position: relative;

margin-top: 12px;

/\*margin-bottom: 16px;\*/

line-height: 32px;

font-size: 28px;

color: #00a1d6;

font-weight: 600;

text-align: center;

}

</style>

<template>

<div class="login-container">

<el-form ref="loginForm" :model="loginForm" :rules="loginRules" class="login-form" auto-complete="on" label-position="left">

<div class="title-container">

<h3 class="title">米果新媒体矩阵管理前端</h3>

</div>

<el-form-item prop="nickname">

<span class="svg-container">

<svg-icon icon-class="user" />

</span>

<el-input

ref="nickname"

v-model="loginForm.nickname"

placeholder="请输入账号"

name="nickname"

type="text"

tabindex="1"

auto-complete="on"

/>

</el-form-item>

<el-form-item prop="password">

<span class="svg-container">

<svg-icon icon-class="password" />

</span>

<el-input

:key="passwordType"

ref="password"

v-model="loginForm.password"

:type="passwordType"

placeholder="请输入密码"

name="password"

tabindex="2"

/>

<span class="show-pwd" @click="showPwd">

<svg-icon :icon-class="passwordType === 'password' ? 'eye' : 'eye-open'" />

</span>

</el-form-item>

<el-button :loading="loading" type="primary" style="width:100%;margin-bottom:30px;" @click="submitForm">Login</el-button>

<div class="tips">

<span style="margin-right:20px;">让世界看到你的价值。</span>

</div>

</el-form>

</div>

</template>

<script>

import LoginApi from '@/api/website/LoginApi'

import { mapGetters } from 'vuex'

export default {

name: "Login",

data() {

return {

loginForm: {

nickname: '',

password: ''

},

loginRules: {

nickname: ,

password:

},

loading: false,

passwordType: 'password',

redirect: undefined,

user:

}

},

computed: {

...mapGetters([

'roles',

])

},

watch: {

$route: {

handler: function(route) {

this.redirect = route.query && route.query.redirect

},

immediate: true

}

},

methods: {

showPwd() {

if (this.passwordType === 'password') {

this.passwordType = ''

} else {

this.passwordType = 'password'

}

this.$nextTick(() => {

this.$refs.password.focus()

})

},

submitForm() {

this.$refs.validate((valid) => {

if(!valid){

this.$message({

message: '信息填写不完整！',

type: 'warning'

})

return false

} else {

this.loading = true

LoginApi.login(this.loginForm).then(data => {

if (data === 'admin' || data === 'staff') {

this.user.push(data)

sessionStorage.setItem('user',data)

this.$store.dispatch('user/getInfo', this.user)

this.$router.push({ path: this.redirect || '/' })

}

else {

this.$message.error('账号或者密码不正确！')

}

this.loading = false

}).catch(err => {

// debugger

console.log(err)

if (this.loginForm.nickname !== '' || this.loginForm.password !== '') {

this.$message.warning('信息填写不完整！')

return false

}

this.loading = false

})

}

})

}

}

}

</script>

<style lang="scss">

/\* 修复input 背景不协调 和光标变色 \*/

/\* Detail see https://github.com/PanJiaChen/vue-element-admin/pull/927 \*/

$bg:#283443;

$light\_gray:#fff;

$cursor: #fff;

@supports (-webkit-mask: none) and (not (cater-color: $cursor)) {

.login-container .el-input input {

color: $cursor;

}

}

/\* reset element-ui css \*/

.login-container {

.el-input {

display: inline-block;

height: 47px;

width: 85%;

input {

background: transparent;

border: 0px;

-webkit-appearance: none;

border-radius: 0px;

padding: 12px 5px 12px 15px;

color: $light\_gray;

height: 47px;

caret-color: $cursor;

&:-webkit-autofill {

box-shadow: 0 0 0px 1000px $bg inset !important;

-webkit-text-fill-color: $cursor !important;

}

}

}

.el-form-item {

border: 1px solid rgba(255, 255, 255, 0.1);

background: rgba(0, 0, 0, 0.1);

border-radius: 5px;

color: #454545;

}

}

</style>

<style lang="scss" scoped>

$bg:#2d3a4b;

$dark\_gray:#889aa4;

$light\_gray:#eee;

.login-container {

min-height: 100%;

width: 100%;

background-color: $bg;

overflow: hidden;

.login-form {

position: relative;

width: 520px;

max-width: 100%;

padding: 160px 35px 0;

margin: 0 auto;

overflow: hidden;

}

.tips {

font-size: 14px;

color: #fff;

margin-bottom: 10px;

span {

&:first-of-type {

margin-right: 16px;

}

}

}

.svg-container {

padding: 6px 5px 6px 15px;

color: $dark\_gray;

vertical-align: middle;

width: 30px;

display: inline-block;

}

.title-container {

position: relative;

.title {

font-size: 26px;

color: $light\_gray;

margin: 0px auto 40px auto;

text-align: center;

font-weight: bold;

}

}

.show-pwd {

position: absolute;

right: 10px;

top: 7px;

font-size: 16px;

color: $dark\_gray;

cursor: pointer;

user-select: none;

}

}

</style>

<template>

</template>

<script>

export default {

name: "index"

}

</script>

<style scoped>

</style>

<template>

<div style="padding: 0 15px;" @click="toggleClick">

<svg

:class="{'is-active':isActive}"

class="hamburger"

viewBox="0 0 1024 1024"

xmlns="http://www.w3.org/2000/svg"

width="64"

height="64"

>

<path d="M408 442h480c4.4 0 8-3.6 8-8v-56c0-4.4-3.6-8-8-8H408c-4.4 0-8 3.6-8 8v56c0 4.4 3.6 8 8 8zm-8 204c0 4.4 3.6 8 8 8h480c4.4 0 8-3.6 8-8v-56c0-4.4-3.6-8-8-8H408c-4.4 0-8 3.6-8 8v56zm504-486H120c-4.4 0-8 3.6-8 8v56c0 4.4 3.6 8 8 8h784c4.4 0 8-3.6 8-8v-56c0-4.4-3.6-8-8-8zm0 632H120c-4.4 0-8 3.6-8 8v56c0 4.4 3.6 8 8 8h784c4.4 0 8-3.6 8-8v-56c0-4.4-3.6-8-8-8zM142.4 642.1L298.7 519a8.84 8.84 0 0 0 0-13.9L142.4 381.9c-5.8-4.6-14.4-.5-14.4 6.9v246.3a8.9 8.9 0 0 0 14.4 7z" />

</svg>

</div>

</template>

<script>

export default {

name: 'Hamburger',

props: {

isActive: {

type: Boolean,

default: false

}

},

methods: {

toggleClick() {

this.$emit('toggleClick')

}

}

}

</script>

<style scoped>

.hamburger {

display: inline-block;

vertical-align: middle;

width: 20px;

height: 20px;

}

.hamburger.is-active {

transform: rotate(180deg);

}

</style>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleCreate">

新增

</el-button>

</div>

<el-table

:key="tableKey"

v-loading="listLoading"

:data="list"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.updateAt }}</span>

</template>

</el-table-column>

<el-table-column label="视频标题" min-width="120px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoTitle }}</span>

</template>

</el-table-column>

<el-table-column label="视频简介" max-width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoProfile }}</span>

</template>

</el-table-column>

<el-table-column label="审核人" width="100px" align="center">

<template slot-scope="{row}">

<span style="color:orange;">{{ row.videoReviewer }}</span>

</template>

</el-table-column>

<el-table-column label="状态" class-name="status-col" width="100px">

<template slot-scope="{row}">

<el-tag :type="row.videoStatus | statusFilter">

{{ row.videoStatus | statusNameFilter}}

</el-tag>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-button v-if="row.videoStatus!=='reviewed' && row.videoStatus!=='reviewing' && row.videoStatus!=='processing' && row.videoStatus!=='complete'" type="primary" size="mini" @click="handleUpdate(row)">

编辑

</el-button>

<el-button v-if="row.videoStatus!=='reviewed' && row.videoStatus!=='reviewing' && row.videoStatus!=='reject' && row.videoStatus!=='processing' && row.videoStatus!=='complete'" size="mini" type="success" @click="handleModifyStatus(row,'reviewing')">

提审

</el-button>

<el-button v-if="row.videoStatus!=='draft' && row.videoStatus!=='reviewed' && row.videoStatus!=='reject' && row.videoStatus!=='processing' && row.videoStatus!=='complete'" size="mini" @click="handleModifyStatus(row,'draft')">

撤回

</el-button>

<el-button v-if="row.videoStatus!=='reviewed' && row.videoStatus!=='reviewing' && row.videoStatus!=='reject' && row.videoStatus!=='processing' && row.videoStatus!=='complete'" size="mini" type="danger" @click="confirmDelete(row)">

删除

</el-button>

<el-tag v-if="row.videoStatus==='reviewed'">

已通过审核，当前权限无法进行其他操作

</el-tag>

<el-tag v-if="row.videoStatus==='complete'" :type="row.videoStatus | statusFilter">

已分发

</el-tag>

<el-tag v-if="row.videoStatus==='processing'" :type="row.videoStatus | statusFilter">

分发中

</el-tag>

</template>

</el-table-column>

</el-table>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<el-dialog :title="textMap" :visible.sync="dialogFormVisible">

<el-form ref="dataForm" :rules="rules" :model="temp" label-position="left" label-width="100px" style="width: 600px; margin-left:50px;">

<el-form-item label="视频标题" prop="videoTitle">

<el-input v-model="temp.videoTitle" />

</el-form-item>

<el-form-item label="创作团队" prop="videoAuthor">

<el-input v-model="temp.videoAuthor" />

</el-form-item>

<el-form-item label="视频简介" prop="videoProfile">

<el-input v-model="temp.videoProfile" :autosize="{ minRows: 2, maxRows: 4}" type="textarea" placeholder="请输入简介" />

</el-form-item>

<el-form-item label="视频封面" prop="videoPic">

<el-button v-show="!temp.videoPic" type="primary" size="mini" @click="imageCropperShow()">

上传图片

</el-button>

<el-button v-show="temp.videoPic" type="primary" size="mini" @click="imageCropperShow()">

更新图片

</el-button>

<el-button v-show="temp.videoPic" type="primary" size="mini" @click="picShow(temp.videoPic)">

预览

</el-button>

</el-form-item>

<el-form-item label="上传视频" prop="videoPath">

<el-upload

:action="this.$store.state.settings.uploadUrl"

:show-file-list="false"

list-type="picture-card"

:on-success="answerVideoSuccess"

:before-upload="beforeUpload"

:on-progress="uploadVideoProcess"

:on-remove="handleRemove"

:on-error="handleError"

accept="video/mp4,video/flv,video/avi,video/wmv"

style="margin-top: 5%;">

<video v-if="answerVideoUrl && videoFlag === false" controls="controls" :src="answerVideoUrl" class="avatar"></video>

<i v-if="!answerVideoUrl && videoFlag === false" class="el-icon-plus avatar-uploader-icon"></i>

<el-progress v-if="videoFlag === true" type="circle" :percentage="percent" style="margin-top: 10px"></el-progress>

</el-upload>

</el-form-item>

<el-form-item label="视频分区" prop="videoClass">

<el-select v-model="temp.videoClass" class="filter-item" placeholder="类目一览" value="">

<el-option v-for="item in videoClassOptions" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</el-form-item>

<el-form-item label="视频标签" prop="videoTag">

<el-tag

:key="tag"

v-for="tag in temp.videoTag"

closable

:disable-transitions="false"

@close="handleClose(tag)">

{{tag}}

</el-tag>

<el-input

class="input-new-tag"

v-if="inputVisible"

v-model="inputValue"

ref="saveTagInput"

size="small"

@keyup.enter.native="handleInputConfirm"

@blur="handleInputConfirm"

>

</el-input>

<el-button v-else class="button-new-tag" size="small" @click="showInput">+ 新标签</el-button>

</el-form-item>

<el-form-item label="视频状态" prop="videoStatus">

<el-select v-model="temp.videoStatus" class="filter-item" placeholder="保存草稿/提交审核" value="">

<el-option v-for="item in statusOptions" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer">

<el-button @click="handleCancel()">

取消

</el-button>

<el-button type="primary" @click="dialogStatus==='create'?createData():updateData()">

确认

</el-button>

</div>

</el-dialog>

<el-dialog title="提示" :visible.sync="delVisible" width="300px" center>

<div class="del-dialog-cnt">删除不可恢复，是否确定删除？</div>

<span slot="footer" class="dialog-footer">

<el-button @click="delVisible = false">取 消</el-button>

<el-button type="primary" @click="handleDelete(row)" >确 定</el-button>

</span>

</el-dialog>

<my-upload

method="POST"

field="file"

v-model="cropperShow"

:width=1200

:height=750

:url="this.$store.state.settings.uploadUrl"

lang-type='zh'

img-format='jpg'

img-bgc='#FFF'

:no-circle=true

@crop-upload-success="cropUploadSuccess">

</my-upload>

<el-dialog title="图片预览" :visible.sync="picVisible" width="600px" center>

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

<img :src="answerPicImageUrl" alt="" class="title-pic">

</div>

<span slot="footer" class="dialog-footer">

<el-button @click="picVisible = false">确定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import MediaVideoApi from "@/api/media/MediaVideoApi";

import 'babel-polyfill'; // es6 shim

import myUpload from 'vue-image-crop-upload';

export default {

name: 'media\_video',

components: { Pagination, myUpload },

directives: { waves },

filters: {

statusNameFilter(status) {

const statusMap = {

reviewing: '审核中',

draft: '草稿',

reviewed: '已审核',

reject: '被驳回',

processing: '分发中',

complete: '分发完毕'

}

return statusMap

},

statusFilter(status) {

const statusMap = {

reviewing: 'warning',

reviewed: 'success',

draft: 'info',

reject: 'danger',

processing: 'warning',

}

return statusMap

}

},

data() {

return {

videoFlag: false,

percent: 0,

inputVisible: false,

inputValue: '',

cropperShow: false,

picVisible: false,

filterIsDel: 1,

delVisible: false,

tableKey: 0,

list: null,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC'

},

// 已过审的不允许删除和存为草稿和提交审核

statusOptions: [{

label: '提交审核',

value: 'reviewing'

}, {

label: '存为草稿',

value: 'draft'

}],

videoClassOptions: [{

label: '日常',

value: '21'

}, {

label: '搞笑',

value: '138'

}, {

label: '美食',

value: '76'

}, {

label: '科普',

value: '124'

}, {

label: '技术',

value: '122'

}, {

label: '动物',

value: '75'

}, {

label: '运动',

value: '163'

},

],

showReviewer: false,

temp: {

videoAuthor: '',

videoProfile: '',

videoTitle: '',

videoStatus: '',

videoUrl: '',

videoReviewer: '',

videoPic: '',

videoClass: '',

videoTag: ,

videoPath: '',

updateAt:''

},

videoTagTemp: undefined,

dialogFormVisible: false,

dialogStatus: '',

textMap: {

update: '编辑',

create: '创建'

},

filterMap: [{

label: '全部',

value: 1

}, {

label: '未删除',

value: 2

}, {

label: '已删除',

value: 3

}],

rules: {

videoTitle: ,

videoProfile: ,

videoClass: ,

videoStatus: ,

videoPic: ,

videoPath: ,

},

downloadLoading: false,

answerPicImageUrl: '',

answerVideoUrl: ''

}

},

created() {

this.getList()

},

methods: {

// 刷新界面

getList() {

this.listLoading = true

MediaVideoApi.findAllByKeywords(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

Object.keys(this.list).forEach(key => (this.list.videoTag = JSON.parse(this.list.videoTag)))

})

this.listLoading = false

},

handleFilter() {

this.listQuery.page = 1

this.getList()

},

handleModifyStatus(row, status) {

this.listLoading = true

row.videoStatus = status

this.temp = Object.assign({}, row) // copy obj

this.videoTagTemp = this.temp.videoTag

this.temp.videoTag = JSON.stringify(this.temp.videoTag)

MediaVideoApi.update(this.temp).then(() => {

for (const v of this.list) {

if (v.id === this.temp.id) {

const index = this.list.indexOf(v)

this.list.splice(index, 1, this.temp)

break

}

}

this.temp.videoTag = this.videoTagTemp

})

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.filterIsDel = 1

this.handleFilter()

},

handleCreate() {

this.answerPicImageUrl = ''

this.answerVideoUrl = ''

Object.keys(this.temp).forEach(key => (this.temp = ''))

this.temp.videoTag =

this.dialogStatus = 'create'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

createData() {

this.$refs.validate((valid) => {

if (valid) {

this.videoTagTemp = this.temp.videoTag

this.dialogFormVisible = false

this.temp.videoTag = JSON.stringify(this.temp.videoTag)

const tempData = Object.assign({}, this.temp)

this.temp.videoTag = this.videoTagTemp

MediaVideoApi.add(tempData).then(() => {

this.list.unshift(this.temp)

this.getList()

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

})

}

})

},

handleUpdate(row) {

this.temp = Object.assign({}, row) // copy obj

if(this.temp.videoPic){

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + this.temp.videoPic

}

if(this.temp.videoPath){

this.answerVideoUrl = this.$store.state.settings.callbackUrl + this.temp.videoPath

}

this.dialogStatus = 'update'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

handleCancel(){

this.dialogFormVisible = false

this.answerPicImageUrl = ''

this.answerVideoUrl = ''

},

updateData() {

this.$refs.validate((valid) => {

if (valid) {

this.videoTagTemp = this.temp.videoTag

this.dialogFormVisible = false

this.temp.videoTag = JSON.stringify(this.temp.videoTag)

const tempData = Object.assign({}, this.temp)

this.temp.videoTag = this.videoTagTemp

MediaVideoApi.update(tempData).then(() => {

this.getList()

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

})

}

})

},

confirmDelete(row) {

// this.delVisible = true

if(confirm('确定要删除吗')===true){

this.handleDelete(row)

}

},

handleDelete(row) {

MediaVideoApi.delOne(row.id).then(data => {

this.getList()

})

this.$notify({

title: 'Success',

message: '成功删除',

type: 'success',

duration: 2000

})

const index = this.list.indexOf(row)

this.list.splice(index, 1)

},

picShow(pic) {

this.picVisible = !this.picVisible

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + pic

},

// 显示图片上传模块

imageCropperShow() {

this.cropperShow = !this.cropperShow

},

// 图片上传成功后执行

cropUploadSuccess(jsonData, field){

this.temp.videoPic = jsonData.data

this.$notify({

title: 'success',

message: '图片上传成功',

type: 'success',

duration: 2000

})

},

handleClose(tag) {

this.temp.videoTag.splice(this.temp.videoTag.indexOf(tag), 1);

},

showInput() {

this.inputVisible = true;

this.$nextTick(\_ => {

this.$refs.saveTagInput.$refs.input.focus();

});

},

handleInputConfirm() {

let inputValue = this.inputValue;

if (inputValue) {

console.log(inputValue)

this.temp.videoTag.push(inputValue);

}

this.inputVisible = false;

this.inputValue = '';

},

beforeUpload (file) {

// 只允许上传2GB以内大小的图片

const isLt2G = file.size / 1024 / 1024 / 1024 < 2;

if (!isLt2G) {

this.$message.error('上传视频大小不能超过2GB!');

}

return isLt2G;

},

uploadVideoProcess(event, file, fileList){

this.videoFlag = true;

this.percent = Math.floor(event.percent)

},

handleRemove(file, fileList) {

this.temp.videoPath = '';

},

answerVideoSuccess(res, file){

this.temp.videoPath = res.data

this.answerVideoUrl = URL.createObjectURL(file.raw)

this.videoFlag = false;

},

handleError(err, file, fileList) {

// 上传失败异常处理

const error = JSON.parse(JSON.stringify(err));

console.log(err)

console.log(error)

this.$message.error(error.status.toString());

this.videoFlag = false;

this.percent = 0;

},

}

}

</script>

<style>

.el-tag + .el-tag {

margin-left: 10px;

}

.button-new-tag {

margin-left: 10px;

height: 32px;

line-height: 30px;

padding-top: 0;

padding-bottom: 0;

}

.input-new-tag {

width: 90px;

margin-left: 10px;

vertical-align: bottom;

}

.title-pic{

width: 90%;

}

.avatar-uploader .el-upload {

border: 1px dashed #d9d9d9;

border-radius: 6px;

cursor: pointer;

position: relative;

overflow: hidden;

width: 178px;

height: 178px;

}

.avatar-uploader .el-upload:hover {

border-color: #409EFF;

}

.avatar-uploader-icon {

line-height: 140px;

}

.avatar {

height: 144px;

}

.image-preview {

width: 178px;

height: 178px;

position: relative;

border: 1px dashed #d9d9d9;

border-radius: 6px;

float: left;

}

.image-preview .image-preview-wrapper {

position: relative;

width: 100%;

height: 100%;

}

.image-preview .image-preview-wrapper img {

width: 100%;

height: 100%;

}

.image-preview .image-preview-action {

position: absolute;

width: 100%;

height: 100%;

left: 0;

top: 0;

cursor: default;

text-align: center;

color: #fff;

opacity: 0;

font-size: 20px;

background-color: rgba(0, 0, 0, .5);

transition: opacity .3s;

cursor: pointer;

text-align: center;

line-height: 200px;

}

.image-preview .image-preview-action .el-icon-delete {

font-size: 32px;

}

.image-preview:hover .image-preview-action {

opacity: 1;

}

.el-upload--picture-card{

display: block;

width: 258px;

height: 146px;

overflow: hidden;

}

</style>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleNewRows">

新增

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="danger" icon="el-icon-delete" @click="confirmDeleteRows" :disabled="this.multipleSelection.length === 0">

批量删除

</el-button>

</div>

<el-table-editabled v-model="tableData" :columns="" ref="editTable">

<el-table

:key="tableKey"

v-loading="listLoading"

:data="tableData"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

@selection-change="handleSelectionChange"

>

<el-table-column type="selection" width="50px" align="center">

</el-table-column>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.updateAt }}</span>

</template>

</el-table-column>

<el-table-column label="活动名称" min-width="80px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="activityName">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.activityName}}</span>

<el-input v-show="rowStates.editing" v-model="row.activityName" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="活动直播链接" min-width="80px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="activityLive">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.activityLive}}</span>

<el-input v-show="rowStates.editing" v-model="row.activityLive" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="活动简介" min-width="200px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="activityProfile">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.activityProfile}}</span>

<el-input v-show="rowStates.editing" v-model="row.activityProfile" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="活动状态" class-name="status-col" width="120px">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="activityActive">

<template slot-scope="{ rowStates, validateOwn }">

<el-select v-show="rowStates.editing" v-model="row.activityActive" class="filter-item" placeholder="启用活动/停用活动" clearable @change="validateOwn">

<el-option v-for="item in statusOptions" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

<el-tag v-show="!rowStates.editing" :type="row.activityActive | statusFilter">

{{ row.activityActive | statusNameFilter}}

</el-tag>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row">

<template slot-scope="{ rowStates }">

<el-button v-if="!rowStates.editing" type="primary" size="mini" @click="editRow(row)">

编辑

</el-button>

<el-button v-if="!rowStates.editing" size="mini" type="danger" @click="delOne(row)">

删除

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" type="success" @click="createRow(row)">

保存

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" @click="createCancelRow(row)">

取消

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" type="success" @click="updateRow(row)">

更新

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" @click="updateCancelRow(row)">

撤销

</el-button>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

</el-table>

</el-table-editabled>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import MpActivityApi from "@/api/miniprogram/MpActivityApi";

export default {

name: 'MpActivity',

components: { Pagination },

directives: { waves },

filters: {

statusFilter(status) {

const statusMap = {

true: 'success',

false: 'danger',

}

return statusMap

},

statusNameFilter(status) {

const statusMap = {

true: '活跃',

false: '已停用',

}

return statusMap

},

},

computed: {

editTable () {

return this.$refs.editTable

}

},

data() {

return {

multipleSelection: ,

isUpdate: true,

delVisible: false,

tableKey: 0,

tableData: ,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC'

},

// 改变活动状态

statusOptions: [{

label: '启用',

value: true

}, {

label: '停用',

value: false

}],

temp: {

activityName: "",

updateAt: '',

activityActive: '',

activityProfile: '',

activityLive: '',

isDel: ''

},

rules: {

videoTitle: ,

videoStatus: ,

},

}

},

created() {

this.getList()

},

methods: {

// 刷新界面

getList() {

this.listLoading = true

MpActivityApi.findAllByKeywords(this.listQuery).then(data => {

this.tableData = data.data

this.total = data.total;

})

this.listLoading = false

},

// 创建选中行的数组

handleSelectionChange (row) {

console.log(row)

this.multipleSelection = row

},

// 每次变动排序后的刷新

handleFilter() {

this.listQuery.page = 1

this.getList()

},

// 触发排序

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

// 回传排序顺序

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.handleFilter()

},

// 点击新增之后的事件

handleNewRows () {

this.isUpdate = false

Object.keys(this.temp).forEach(key => (this.temp = ''))

const newRow = this.temp

this.editTable.newRows()

newRow.isDel = true

},

// 点击新增、再点击保存之后的事件，用于返回新增数据，需调用add方法

createRow (row) {

this.$refs.editTable.validateRows(, valid => {

// valid 为布尔值，代表表格正在编辑的所有字段是否验证通过

if (valid) {

const tempData = Object.assign({}, row)

MpActivityApi.add(tempData).then(() => {

// this.tableData.unshift(tempData)

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击编辑之后的状态，使某行数据可编辑

editRow (row) {

this.editTable.editRows()

},

// 点击更新之后，将数据上传到后端，并且更新前端数据

updateRow (row) {

this.editTable.validateRows(,valid => {

if (valid) {

const tempData = Object.assign({}, row)

MpActivityApi.update(tempData).then(() => {

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击撤销之后，还原目标行状态

updateCancelRow (row) {

this.editTable.cancelRows()

},

// 点击取消后，删除对应空行

createCancelRow (row) {

this.editTable.delRows()

},

// 删除二次确认

confirmDeleteRows () {

if(confirm('确定要删除吗')===true){

this.deleteGroup()

}

},

// 删除选中数据

deleteGroup () {

MpActivityApi.del(this.multipleSelection).then(data => {

console.log(data)

this.getList()

})

},

// 删除某行数据（无需确认）

delOne (row) {

console.log(row)

this.multipleSelection.push(row)

MpActivityApi.del(this.multipleSelection).then(data => {

console.log(data)

this.getList()

})

}

}

}

</script>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleNewRows">

新增

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="danger" icon="el-icon-delete" @click="confirmDeleteRows" :disabled="this.multipleSelection.length === 0">

批量删除

</el-button>

</div>

<el-table-editabled v-model="tableData" :columns="" ref="editTable">

<el-table

:key="tableKey"

v-loading="listLoading"

:data="tableData"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

@selection-change="handleSelectionChange"

>

<el-table-column type="selection" width="50px" align="center">

</el-table-column>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.updateAt }}</span>

</template>

</el-table-column>

<el-table-column label="小组名称" min-width="80px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="groupName">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.groupName}}</span>

<el-input v-show="rowStates.editing" v-model="row.groupName" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="小组简介" min-width="200px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="groupProfile">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.groupProfile}}</span>

<el-input v-show="rowStates.editing" v-model="row.groupProfile" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="小组获票数" min-width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.groupVotes }}</span>

</template>

</el-table-column>

<el-table-column label="小组图片" min-width="100px" align="center">

<template slot-scope="{row}">

<el-button v-show="!row.groupPicHd" type="primary" size="mini" @click="imageCropperShow(row)">

上传图片

</el-button>

<el-button v-show="row.groupPicHd" type="primary" size="mini" @click="imageCropperShow(row)">

更新图片

</el-button>

<el-button v-show="row.groupPicHd" type="primary" size="mini" @click="picShow(row.groupPicHd)">

预览

</el-button>

</template>

</el-table-column>

<el-table-column label="小组对应活动" min-width="100px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="activityId">

<template slot-scope="{ rowStates, validateOwn }">

<el-select

v-show="rowStates.editing"

v-model="row.activityId"

class="filter-item"

filterable

remote

reserve-keyword

clearable

placeholder="请选择对应活动"

:remote-method="remoteMethod"

:loading="loading"

@change="validateOwn">

<el-option

v-for="item in options"

:key="item.id"

:label="item.activityName"

:value="item.id">

</el-option>

</el-select>

<span v-show="!rowStates.editing">{{row.activityName}}</span>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row">

<template slot-scope="{ rowStates }">

<el-button v-if="!rowStates.editing" type="primary" size="mini" @click="editRow(row)">

编辑

</el-button>

<el-button v-if="!rowStates.editing" size="mini" type="danger" @click="delOne(row)">

删除

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" type="success" @click="createRow(row)">

保存

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" @click="createCancelRow(row)">

取消

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" type="success" @click="updateRow(row)">

更新

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" @click="updateCancelRow(row)">

撤销

</el-button>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

</el-table>

</el-table-editabled>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<my-upload

method="POST"

field="file"

v-model="cropperShow"

:width=640

:height=360

:url="this.$store.state.settings.mpUploadUrl"

lang-type='zh'

img-format='jpg'

img-bgc='#FFF'

:no-circle=true

@crop-upload-success="cropUploadSuccess">

</my-upload>

<el-dialog title="图片预览" :visible.sync="picVisible" width="720px" center>

<h3>

大图

</h3>

<img :src="answerPicImageUrl">

<h3>

小图

</h3>

<img :src="mpAnswerPicImageUrl">

<span slot="footer" class="dialog-footer">

<el-button @click="picVisible = false">确定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import MpGroupApi from "@/api/miniprogram/MpGroupApi";

import MpActivityApi from "@/api/miniprogram/MpActivityApi";

import 'babel-polyfill'; // es6 shim

import myUpload from 'vue-image-crop-upload';

export default {

name: 'MpGroup',

components: { Pagination, myUpload },

directives: { waves },

filters: {

statusFilter(status) {

const statusMap = {

true: 'success',

false: 'danger',

}

return statusMap

},

statusNameFilter(status) {

const statusMap = {

true: '活跃',

false: '已停用',

}

return statusMap

},

},

computed: {

editTable () {

return this.$refs.editTable

}

},

data() {

return {

answerPicImageUrl: "",

mpAnswerPicImageUrl: "",

picVisible: false,

loading: false,

cropperShow: false,

multipleSelection: ,

isUpdate: true,

tableKey: 0,

tableData: ,

options: ,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC'

},

// 改变活动状态

statusOptions: [{

label: '启用',

value: true

}, {

label: '停用',

value: false

}],

temp: {

groupName: "",

updateAt: '',

groupProfile: '',

groupVotes: '',

groupPic: '',

groupPicHd: '',

isDel: '',

activityId: '',

activityName: ''

},

rules: {

videoTitle: ,

videoStatus: ,

},

}

},

created() {

this.getList()

},

methods: {

// 选择活动时，用于根据关键字查找活动

remoteMethod (query) {

if (query === '') {

this.options =

return

}

this.loading = true

MpActivityApi.findAllByKeywordsFromInput(query).then(data => {

this.loading = false

this.options = data

})

},

// 显示图片上传模块

imageCropperShow(row) {

this.cropperShow = !this.cropperShow

this.temp = Object.assign({}, row) // copy obj

},

// 图片上传成功后执行

cropUploadSuccess(jsonData, field){

this.temp.groupPicHd = jsonData.data

const tempData = Object.assign({}, this.temp)

MpGroupApi.update(tempData).then(() => {

this.$notify({

title: 'success',

message: '图片上传成功',

type: 'success',

duration: 2000

})

this.getList()

})

},

picShow(pic) {

this.picVisible = !this.picVisible

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + pic

this.mpAnswerPicImageUrl = this.$store.state.settings.mpCallbackUrl + pic

},

// 刷新界面

getList() {

this.listLoading = true

MpGroupApi.findAllByKeywords(this.listQuery).then(data => {

this.tableData = data.data

this.total = data.total;

})

this.listLoading = false

},

// 创建选中行的数组

handleSelectionChange (row) {

this.multipleSelection = row

},

// 每次变动排序后的刷新

handleFilter() {

this.listQuery.page = 1

this.getList()

},

// 触发排序

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

// 回传排序顺序

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.handleFilter()

},

// 点击新增之后的事件

handleNewRows () {

this.isUpdate = false

Object.keys(this.temp).forEach(key => (this.temp = ''))

const newRow = this.temp

this.editTable.newRows()

newRow.isDel = true

},

// 点击新增、再点击保存之后的事件，用于返回新增数据，需调用add方法

createRow (row) {

this.$refs.editTable.validateRows(, valid => {

// valid 为布尔值，代表表格正在编辑的所有字段是否验证通过

if (valid) {

const tempData = Object.assign({}, row)

MpGroupApi.add(tempData).then(() => {

// this.tableData.unshift(tempData)

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击编辑之后的状态，使某行数据可编辑

editRow (row) {

if(row.activityId !== ""){

MpActivityApi.findOne(row.activityId).then(data => {

this.options =

this.options.push(data)

})

}

this.editTable.editRows()

},

// 点击更新之后，将数据上传到后端，并且更新前端数据

updateRow (row) {

this.editTable.validateRows(,valid => {

if (valid) {

const tempData = Object.assign({}, row)

MpGroupApi.update(tempData).then(() => {

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击撤销之后，还原目标行状态

updateCancelRow (row) {

this.editTable.cancelRows()

},

// 点击取消后，删除对应空行

createCancelRow (row) {

this.editTable.delRows()

},

// 删除二次确认

confirmDeleteRows () {

if(confirm('确定要删除吗')===true){

this.deleteGroup()

}

},

// 删除选中数据

deleteGroup () {

MpGroupApi.del(this.multipleSelection).then(data => {

console.log(data)

this.getList()

})

},

// 删除某行数据（无需确认）

delOne (row) {

console.log(row)

this.multipleSelection.push(row)

MpGroupApi.del(this.multipleSelection).then(data => {

console.log(data)

this.getList()

})

}

}

}

</script>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleNewRows">

新增

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="danger" icon="el-icon-delete" @click="confirmDeleteRows" :disabled="this.multipleSelection.length === 0">

批量删除

</el-button>

</div>

<el-table-editabled v-model="tableData" :columns="" ref="editTable">

<el-table

:key="tableKey"

v-loading="listLoading"

:data="tableData"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

@selection-change="handleSelectionChange"

>

<el-table-column type="selection" width="50px" align="center">

</el-table-column>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.updateAt }}</span>

</template>

</el-table-column>

<el-table-column label="赞助商名称" min-width="80px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="merchantName">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.merchantName}}</span>

<el-input v-show="rowStates.editing" v-model="row.merchantName" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="赞助商信息" min-width="200px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="merchantInfo">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.merchantInfo}}</span>

<el-input v-show="rowStates.editing" v-model="row.merchantInfo" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="赞助商图标" min-width="100px" align="center">

<template slot-scope="{row}">

<el-button v-show="!row.merchantLogo" type="primary" size="mini" @click="imageCropperShow(row)">

上传图片

</el-button>

<el-button v-show="row.merchantLogo" type="primary" size="mini" @click="imageCropperShow(row)">

更新图片

</el-button>

<el-button v-show="row.merchantLogo" type="primary" size="mini" @click="picShow(row.merchantLogo)">

预览

</el-button>

</template>

</el-table-column>

<el-table-column label="赞助商对应活动" min-width="100px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="activityId">

<template slot-scope="{ rowStates, validateOwn }">

<el-select

v-show="rowStates.editing"

v-model="row.activityId"

class="filter-item"

filterable

remote

reserve-keyword

clearable

placeholder="请选择对应活动"

:remote-method="remoteMethod"

:loading="loading"

@change="validateOwn">

<el-option

v-for="item in options"

:key="item.id"

:label="item.activityName"

:value="item.id">

</el-option>

</el-select>

<span v-show="!rowStates.editing">{{row.activityName}}</span>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row">

<template slot-scope="{ rowStates }">

<el-button v-if="!rowStates.editing" type="primary" size="mini" @click="editRow(row)">

编辑

</el-button>

<el-button v-if="!rowStates.editing" size="mini" type="danger" @click="delOne(row)">

删除

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" type="success" @click="createRow(row)">

保存

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" @click="createCancelRow(row)">

取消

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" type="success" @click="updateRow(row)">

更新

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" @click="updateCancelRow(row)">

撤销

</el-button>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

</el-table>

</el-table-editabled>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<my-upload

method="POST"

field="file"

v-model="cropperShow"

:width=150

:height=150

:url="this.$store.state.settings.uploadUrl"

lang-type='zh'

img-format='jpg'

img-bgc='#FFF'

:no-circle=true

@crop-upload-success="cropUploadSuccess">

</my-upload>

<el-dialog title="图片预览" :visible.sync="picVisible" width="320px" center>

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

<img :src="answerPicImageUrl" alt="">

</div>

<span slot="footer" class="dialog-footer">

<el-button @click="picVisible = false">确定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import MpMerchantApi from "@/api/miniprogram/MpMerchantApi";

import MpActivityApi from "@/api/miniprogram/MpActivityApi";

import 'babel-polyfill'; // es6 shim

import myUpload from 'vue-image-crop-upload';

export default {

name: 'MpMerchant',

components: { Pagination, myUpload },

directives: { waves },

filters: {

statusFilter(status) {

const statusMap = {

true: 'success',

false: 'danger',

}

return statusMap

},

statusNameFilter(status) {

const statusMap = {

true: '活跃',

false: '已停用',

}

return statusMap

},

},

computed: {

editTable () {

return this.$refs.editTable

}

},

data() {

return {

answerPicImageUrl: "",

picVisible: false,

loading: false,

cropperShow: false,

multipleSelection: ,

isUpdate: true,

tableKey: 0,

tableData: ,

options: ,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC'

},

// 改变活动状态

statusOptions: [{

label: '启用',

value: true

}, {

label: '停用',

value: false

}],

temp: {

merchantName: "",

updateAt: '',

merchantInfo: '',

merchantLogo: '',

isDel: '',

activityId: '',

activityName: ''

},

rules: {

videoTitle: ,

videoStatus: ,

},

}

},

created() {

this.getList()

},

methods: {

// 选择活动时，用于根据关键字查找活动

remoteMethod (query) {

if (query === '') {

this.options =

return

}

this.loading = true

MpActivityApi.findAllByKeywordsFromInput(query).then(data => {

this.loading = false

this.options = data

})

},

// 显示图片上传模块

imageCropperShow(row) {

this.cropperShow = !this.cropperShow

this.temp = Object.assign({}, row) // copy obj

},

// 图片上传成功后执行

cropUploadSuccess(jsonData, field){

this.temp.merchantLogo = jsonData.data

const tempData = Object.assign({}, this.temp)

MpMerchantApi.update(tempData).then(() => {

this.$notify({

title: 'success',

message: '图片上传成功',

type: 'success',

duration: 2000

})

this.getList()

})

},

picShow(pic) {

this.picVisible = !this.picVisible

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + pic

},

// 刷新界面

getList() {

this.listLoading = true

MpMerchantApi.findAllByKeywords(this.listQuery).then(data => {

this.tableData = data.data

this.total = data.total;

})

this.listLoading = false

},

// 创建选中行的数组

handleSelectionChange (row) {

this.multipleSelection = row

},

// 每次变动排序后的刷新

handleFilter() {

this.listQuery.page = 1

this.getList()

},

// 触发排序

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

// 回传排序顺序

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.handleFilter()

},

// 点击新增之后的事件

handleNewRows () {

this.isUpdate = false

Object.keys(this.temp).forEach(key => (this.temp = ''))

const newRow = this.temp

this.editTable.newRows()

newRow.isDel = true

},

// 点击新增、再点击保存之后的事件，用于返回新增数据，需调用add方法

createRow (row) {

this.$refs.editTable.validateRows(, valid => {

// valid 为布尔值，代表表格正在编辑的所有字段是否验证通过

if (valid) {

const tempData = Object.assign({}, row)

MpMerchantApi.add(tempData).then(() => {

// this.tableData.unshift(tempData)

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击编辑之后的状态，使某行数据可编辑

editRow (row) {

if(row.activityId !== ""){

MpActivityApi.findOne(row.activityId).then(data => {

this.options =

this.options.push(data)

})

}

this.editTable.editRows()

},

// 点击更新之后，将数据上传到后端，并且更新前端数据

updateRow (row) {

this.editTable.validateRows(,valid => {

if (valid) {

const tempData = Object.assign({}, row)

MpMerchantApi.update(tempData).then(() => {

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击撤销之后，还原目标行状态

updateCancelRow (row) {

this.editTable.cancelRows()

},

// 点击取消后，删除对应空行

createCancelRow (row) {

this.editTable.delRows()

},

// 删除二次确认

confirmDeleteRows () {

if(confirm('确定要删除吗')===true){

this.deleteGroup()

}

},

// 删除选中数据

deleteGroup () {

MpMerchantApi.del(this.multipleSelection).then(data => {

console.log(this.multipleSelection)

this.getList()

})

},

// 删除某行数据（无需确认）

delOne (row) {

console.log(row)

this.multipleSelection.push(row)

MpMerchantApi.del(this.multipleSelection).then(data => {

console.log(data)

this.getList()

})

}

}

}

</script>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleNewRows">

新增

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="danger" icon="el-icon-delete" @click="confirmDeleteRows" :disabled="this.multipleSelection.length === 0">

批量删除

</el-button>

</div>

<el-table-editabled v-model="tableData" :columns="" ref="editTable">

<el-table

:key="tableKey"

v-loading="listLoading"

:data="tableData"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

@selection-change="handleSelectionChange"

>

<el-table-column type="selection" width="50px" align="center">

</el-table-column>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.updateAt }}</span>

</template>

</el-table-column>

<el-table-column label="公告项目名称" min-width="80px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="noteName">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.noteName}}</span>

<el-input v-show="rowStates.editing" v-model="row.noteName" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="公告正文" min-width="200px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="noteDetails">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.noteDetails}}</span>

<el-input v-show="rowStates.editing" v-model="row.noteDetails" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="激活时间" min-width="200px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="noteDisplayTime">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.noteDisplayTime | format }}</span>

<el-date-picker

v-show="rowStates.editing"

v-model="row.noteDisplayTime"

type="datetime"

value-format="timestamp"

@input="validateOwn"

placeholder="选择日期时间">

</el-date-picker>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="赞助商对应活动" min-width="100px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="activityId">

<template slot-scope="{ rowStates, validateOwn }">

<el-select

v-show="rowStates.editing"

v-model="row.activityId"

class="filter-item"

filterable

remote

reserve-keyword

clearable

placeholder="请选择对应活动"

:remote-method="remoteMethod"

:loading="loading"

@change="validateOwn">

<el-option

v-for="item in options"

:key="item.id"

:label="item.activityName"

:value="item.id">

</el-option>

</el-select>

<span v-show="!rowStates.editing">{{row.activityName}}</span>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row">

<template slot-scope="{ rowStates }">

<el-button v-if="!rowStates.editing" type="primary" size="mini" @click="editRow(row)">

编辑

</el-button>

<el-button v-if="!rowStates.editing" size="mini" type="danger" @click="delOne(row)">

删除

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" type="success" @click="createRow(row)">

保存

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" @click="createCancelRow(row)">

取消

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" type="success" @click="updateRow(row)">

更新

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" @click="updateCancelRow(row)">

撤销

</el-button>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

</el-table>

</el-table-editabled>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import MpNoteApi from "@/api/miniprogram/MpNoteApi";

import MpActivityApi from "@/api/miniprogram/MpActivityApi";

import 'babel-polyfill'; // es6 shim

import { formatDate } from '@/utils/timeformat.js';

export default {

name: 'MpGroup',

components: { Pagination },

directives: { waves },

filters: {

format: (value) => formatDate('yyyy-MM-dd hh:mm:ss',new Date(value)),

statusFilter(status) {

const statusMap = {

true: 'success',

false: 'danger',

}

return statusMap

},

statusNameFilter(status) {

const statusMap = {

true: '活跃',

false: '已停用',

}

return statusMap

},

},

computed: {

editTable () {

return this.$refs.editTable

}

},

data() {

return {

answerPicImageUrl: "",

picVisible: false,

loading: false,

cropperShow: false,

multipleSelection: ,

isUpdate: true,

tableKey: 0,

tableData: ,

options: ,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC'

},

// 改变活动状态

statusOptions: [{

label: '启用',

value: true

}, {

label: '停用',

value: false

}],

temp: {

noteName: "",

updateAt: '',

noteDetails: '',

noteDisplayTime: '',

isDel: '',

activityId: '',

activityName: ''

},

rules: {

videoTitle: ,

videoStatus: ,

},

}

},

created() {

this.getList()

},

methods: {

// 选择活动时，用于根据关键字查找活动

remoteMethod (query) {

if (query === '') {

this.options =

return

}

this.loading = true

MpActivityApi.findAllByKeywordsFromInput(query).then(data => {

this.loading = false

this.options = data

})

},

// 刷新界面

getList() {

this.listLoading = true

MpNoteApi.findAllByKeywords(this.listQuery).then(data => {

this.tableData = data.data

this.total = data.total;

})

this.listLoading = false

},

// 创建选中行的数组

handleSelectionChange (row) {

this.multipleSelection = row

},

// 每次变动排序后的刷新

handleFilter() {

this.listQuery.page = 1

this.getList()

},

// 触发排序

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

// 回传排序顺序

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.handleFilter()

},

// 点击新增之后的事件

handleNewRows () {

this.isUpdate = false

Object.keys(this.temp).forEach(key => (this.temp = ''))

const newRow = this.temp

this.editTable.newRows()

newRow.isDel = true

},

// 点击新增、再点击保存之后的事件，用于返回新增数据，需调用add方法

createRow (row) {

this.$refs.editTable.validateRows(, valid => {

// valid 为布尔值，代表表格正在编辑的所有字段是否验证通过

if (valid) {

const tempData = Object.assign({}, row)

MpNoteApi.add(tempData).then(() => {

// this.tableData.unshift(tempData)

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击编辑之后的状态，使某行数据可编辑

editRow (row) {

if(row.activityId !== ""){

MpActivityApi.findOne(row.activityId).then(data => {

this.options =

this.options.push(data)

})

}

this.editTable.editRows()

},

// 点击更新之后，将数据上传到后端，并且更新前端数据

updateRow (row) {

this.editTable.validateRows(,valid => {

if (valid) {

const tempData = Object.assign({}, row)

MpNoteApi.update(tempData).then(() => {

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击撤销之后，还原目标行状态

updateCancelRow (row) {

this.editTable.cancelRows()

},

// 点击取消后，删除对应空行

createCancelRow (row) {

this.editTable.delRows()

},

// 删除二次确认

confirmDeleteRows () {

if(confirm('确定要删除吗')===true){

this.deleteGroup()

}

},

// 删除选中数据

deleteGroup () {

MpNoteApi.del(this.multipleSelection).then(data => {

console.log(data)

this.getList()

})

},

// 删除某行数据（无需确认）

delOne (row) {

console.log(row)

this.multipleSelection.push(row)

MpNoteApi.del(this.multipleSelection).then(data => {

console.log(data)

this.getList()

})

}

}

}

</script>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleNewRows">

新增

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="danger" icon="el-icon-delete" @click="confirmDeleteRows" :disabled="this.multipleSelection.length === 0">

批量删除

</el-button>

</div>

<el-table-editabled v-model="tableData" :columns="" ref="editTable">

<el-table

:key="tableKey"

v-loading="listLoading"

:data="tableData"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

@selection-change="handleSelectionChange"

>

<el-table-column type="selection" width="50px" align="center">

</el-table-column>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.updateAt }}</span>

</template>

</el-table-column>

<el-table-column label="投票用户名" min-width="80px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="recordNickname">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.recordNickname}}</span>

<el-input v-show="rowStates.editing" v-model="row.recordNickname" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="投票用户openid" min-width="200px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="recordOpenid">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.recordOpenid}}</span>

<el-input v-show="rowStates.editing" v-model="row.recordOpenid" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="投票时间" min-width="200px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="recordDate">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.recordDate | format }}</span>

<el-date-picker

v-show="rowStates.editing"

v-model="row.recordDate"

type="datetime"

value-format="timestamp"

@input="validateOwn"

placeholder="选择日期时间">

</el-date-picker>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="投票对象" min-width="100px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="groupId">

<template slot-scope="{ rowStates, validateOwn }">

<el-select

v-show="rowStates.editing"

v-model="row.groupId"

class="filter-item"

filterable

remote

reserve-keyword

clearable

placeholder="请选择对应活动"

:remote-method="remoteMethod"

:loading="loading"

@change="validateOwn">

<el-option

v-for="item in options"

:key="item.id"

:label="item.groupName"

:value="item.id">

</el-option>

</el-select>

<span v-show="!rowStates.editing">{{row.groupName}}</span>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row">

<template slot-scope="{ rowStates }">

<el-button v-if="!rowStates.editing" type="primary" size="mini" @click="editRow(row)">

编辑

</el-button>

<el-button v-if="!rowStates.editing" size="mini" type="danger" @click="delOne(row)">

删除

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" type="success" @click="createRow(row)">

保存

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" @click="createCancelRow(row)">

取消

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" type="success" @click="updateRow(row)">

更新

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" @click="updateCancelRow(row)">

撤销

</el-button>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

</el-table>

</el-table-editabled>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import MpRecordApi from "@/api/miniprogram/MpRecordApi";

import MpGroupApi from "@/api/miniprogram/MpGroupApi";

import 'babel-polyfill'; // es6 shim

import { formatDate } from '@/utils/timeformat.js';

export default {

name: 'MpRecord',

components: { Pagination },

directives: { waves },

filters: {

format: (value) => formatDate('yyyy-MM-dd hh:mm:ss',new Date(value)),

statusFilter(status) {

const statusMap = {

true: 'success',

false: 'danger',

}

return statusMap

},

statusNameFilter(status) {

const statusMap = {

true: '活跃',

false: '已停用',

}

return statusMap

},

},

computed: {

editTable () {

return this.$refs.editTable

}

},

data() {

return {

loading: false,

multipleSelection: ,

isUpdate: true,

tableKey: 0,

tableData: ,

options: ,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC'

},

// 改变活动状态

statusOptions: [{

label: '启用',

value: true

}, {

label: '停用',

value: false

}],

temp: {

recordNickname: "",

updateAt: '',

recordOpenid: '',

recordDate: '',

isDel: '',

groupId: '',

groupName: ''

},

rules: {

videoTitle: ,

videoStatus: ,

},

}

},

created() {

this.getList()

},

methods: {

// 选择活动时，用于根据关键字查找活动

remoteMethod (query) {

if (query === '') {

this.options =

return

}

this.loading = true

MpGroupApi.findAllByKeywordsFromInput(query).then(data => {

this.loading = false

this.options = data

})

},

// 刷新界面

getList() {

this.listLoading = true

MpRecordApi.findAllByKeywords(this.listQuery).then(data => {

this.tableData = data.data

this.total = data.total;

})

this.listLoading = false

},

// 创建选中行的数组

handleSelectionChange (row) {

this.multipleSelection = row

},

// 每次变动排序后的刷新

handleFilter() {

this.listQuery.page = 1

this.getList()

},

// 触发排序

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

// 回传排序顺序

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.handleFilter()

},

// 点击新增之后的事件

handleNewRows () {

this.isUpdate = false

Object.keys(this.temp).forEach(key => (this.temp = ''))

const newRow = this.temp

this.editTable.newRows()

newRow.isDel = true

},

// 点击新增、再点击保存之后的事件，用于返回新增数据，需调用add方法

createRow (row) {

this.$refs.editTable.validateRows(, valid => {

// valid 为布尔值，代表表格正在编辑的所有字段是否验证通过

if (valid) {

const tempData = Object.assign({}, row)

MpRecordApi.add(tempData).then(() => {

// this.tableData.unshift(tempData)

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击编辑之后的状态，使某行数据可编辑

editRow (row) {

if(row.groupId !== ""){

MpGroupApi.findOne(row.groupId).then(data => {

this.options =

this.options.push(data)

})

}

this.editTable.editRows()

},

// 点击更新之后，将数据上传到后端，并且更新前端数据

updateRow (row) {

this.editTable.validateRows(,valid => {

if (valid) {

const tempData = Object.assign({}, row)

MpRecordApi.update(tempData).then(() => {

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击撤销之后，还原目标行状态

updateCancelRow (row) {

this.editTable.cancelRows()

},

// 点击取消后，删除对应空行

createCancelRow (row) {

this.editTable.delRows()

},

// 删除二次确认

confirmDeleteRows () {

if(confirm('确定要删除吗')===true){

this.deleteGroup()

}

},

// 删除选中数据

deleteGroup () {

MpRecordApi.del(this.multipleSelection).then(data => {

console.log(data)

this.getList()

})

},

// 删除某行数据（无需确认）

delOne (row) {

console.log(row)

this.multipleSelection.push(row)

MpRecordApi.del(this.multipleSelection).then(data => {

console.log(data)

this.getList()

})

}

}

}

</script>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleNewRows">

新增

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="danger" icon="el-icon-delete" @click="confirmDeleteRows" :disabled="this.multipleSelection.length === 0">

批量删除

</el-button>

</div>

<el-table-editabled v-model="tableData" :columns="" ref="editTable">

<el-table

:key="tableKey"

v-loading="listLoading"

:data="tableData"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

@selection-change="handleSelectionChange"

>

<el-table-column type="selection" width="50px" align="center">

</el-table-column>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.updateAt }}</span>

</template>

</el-table-column>

<el-table-column label="轮播图名称" min-width="80px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="swiperName">

<template slot-scope="{ rowStates, validateOwn }">

<span v-show="!rowStates.editing">{{row.swiperName}}</span>

<el-input v-show="rowStates.editing" v-model="row.swiperName" clearable @input="validateOwn">

</el-input>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="轮播图" min-width="100px" align="center">

<template slot-scope="{row}">

<el-button v-show="!row.swiperPic" type="primary" size="mini" @click="imageCropperShow(row)">

上传图片

</el-button>

<el-button v-show="row.swiperPic" type="primary" size="mini" @click="imageCropperShow(row)">

更新图片

</el-button>

<el-button v-show="row.swiperPic" type="primary" size="mini" @click="picShow(row.swiperPic)">

预览

</el-button>

</template>

</el-table-column>

<el-table-column label="轮播图对应活动" min-width="100px" align="center">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row" prop="activityId">

<template slot-scope="{ rowStates, validateOwn }">

<el-select

v-show="rowStates.editing"

v-model="row.activityId"

class="filter-item"

filterable

remote

reserve-keyword

clearable

placeholder="请选择对应活动"

:remote-method="remoteMethod"

:loading="loading"

@change="validateOwn">

<el-option

v-for="item in options"

:key="item.id"

:label="item.activityName"

:value="item.id">

</el-option>

</el-select>

<span v-show="!rowStates.editing">{{row.activityName}}</span>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-table-editabled-cell :row="row">

<template slot-scope="{ rowStates }">

<el-button v-if="!rowStates.editing" type="primary" size="mini" @click="editRow(row)">

编辑

</el-button>

<el-button v-if="!rowStates.editing" size="mini" type="danger" @click="delOne(row)">

删除

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" type="success" @click="createRow(row)">

保存

</el-button>

<el-button v-if="rowStates.editing && row.isDel" size="mini" @click="createCancelRow(row)">

取消

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" type="success" @click="updateRow(row)">

更新

</el-button>

<el-button v-if="rowStates.editing && !row.isDel" size="mini" @click="updateCancelRow(row)">

撤销

</el-button>

</template>

</el-table-editabled-cell>

</template>

</el-table-column>

</el-table>

</el-table-editabled>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<my-upload

method="POST"

field="file"

v-model="cropperShow"

:width=720

:height=240

:url="this.$store.state.settings.uploadUrl"

lang-type='zh'

img-format='jpg'

img-bgc='#FFF'

:no-circle=true

@crop-upload-success="cropUploadSuccess">

</my-upload>

<el-dialog title="图片预览" :visible.sync="picVisible" width="960px" center>

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

<img :src="answerPicImageUrl" alt="">

</div>

<span slot="footer" class="dialog-footer">

<el-button @click="picVisible = false">确定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import MpSwiperApi from "@/api/miniprogram/MpSwiperApi";

import MpActivityApi from "@/api/miniprogram/MpActivityApi";

import 'babel-polyfill'; // es6 shim

import myUpload from 'vue-image-crop-upload';

export default {

name: 'MpGroup',

components: { Pagination, myUpload },

directives: { waves },

filters: {

statusFilter(status) {

const statusMap = {

true: 'success',

false: 'danger',

}

return statusMap

},

statusNameFilter(status) {

const statusMap = {

true: '活跃',

false: '已停用',

}

return statusMap

},

},

computed: {

editTable () {

return this.$refs.editTable

}

},

data() {

return {

answerPicImageUrl: "",

mpAnswerPicImageUrl: "",

picVisible: false,

loading: false,

cropperShow: false,

multipleSelection: ,

isUpdate: true,

tableKey: 0,

tableData: ,

options: ,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC'

},

// 改变活动状态

statusOptions: [{

label: '启用',

value: true

}, {

label: '停用',

value: false

}],

temp: {

swiperName: "",

updateAt: '',

swiperPic: '',

isDel: '',

activityId: '',

activityName: ''

},

rules: {

videoTitle: ,

videoStatus: ,

},

}

},

created() {

this.getList()

},

methods: {

// 选择活动时，用于根据关键字查找活动

remoteMethod (query) {

if (query === '') {

this.options =

return

}

this.loading = true

MpActivityApi.findAllByKeywordsFromInput(query).then(data => {

this.loading = false

this.options = data

})

},

// 显示图片上传模块

imageCropperShow(row) {

this.cropperShow = !this.cropperShow

this.temp = Object.assign({}, row) // copy obj

},

// 图片上传成功后执行

cropUploadSuccess(jsonData, field){

this.temp.swiperPic = jsonData.data

const tempData = Object.assign({}, this.temp)

MpSwiperApi.update(tempData).then(() => {

this.$notify({

title: 'success',

message: '图片上传成功',

type: 'success',

duration: 2000

})

this.getList()

})

},

picShow(pic) {

this.picVisible = !this.picVisible

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + pic

},

// 刷新界面

getList() {

this.listLoading = true

MpSwiperApi.findAllByKeywords(this.listQuery).then(data => {

this.tableData = data.data

this.total = data.total;

})

this.listLoading = false

},

// 创建选中行的数组

handleSelectionChange (row) {

this.multipleSelection = row

},

// 每次变动排序后的刷新

handleFilter() {

this.listQuery.page = 1

this.getList()

},

// 触发排序

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

// 回传排序顺序

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.handleFilter()

},

// 点击新增之后的事件

handleNewRows () {

this.isUpdate = false

Object.keys(this.temp).forEach(key => (this.temp = ''))

const newRow = this.temp

this.editTable.newRows()

newRow.isDel = true

},

// 点击新增、再点击保存之后的事件，用于返回新增数据，需调用add方法

createRow (row) {

this.$refs.editTable.validateRows(, valid => {

// valid 为布尔值，代表表格正在编辑的所有字段是否验证通过

if (valid) {

const tempData = Object.assign({}, row)

MpSwiperApi.add(tempData).then(() => {

// this.tableData.unshift(tempData)

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击编辑之后的状态，使某行数据可编辑

editRow (row) {

if(row.activityId !== ""){

MpActivityApi.findOne(row.activityId).then(data => {

this.options =

this.options.push(data)

})

}

this.editTable.editRows()

},

// 点击更新之后，将数据上传到后端，并且更新前端数据

updateRow (row) {

this.editTable.validateRows(,valid => {

if (valid) {

const tempData = Object.assign({}, row)

MpSwiperApi.update(tempData).then(() => {

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

this.getList()

})

}

})

},

// 点击撤销之后，还原目标行状态

updateCancelRow (row) {

this.editTable.cancelRows()

},

// 点击取消后，删除对应空行

createCancelRow (row) {

this.editTable.delRows()

},

// 删除二次确认

confirmDeleteRows () {

if(confirm('确定要删除吗')===true){

this.deleteGroup()

}

},

// 删除选中数据

deleteGroup () {

MpSwiperApi.del(this.multipleSelection).then(data => {

console.log(data)

this.getList()

})

},

// 删除某行数据（无需确认）

delOne (row) {

console.log(row)

this.multipleSelection.push(row)

MpSwiperApi.del(this.multipleSelection).then(data => {

console.log(data)

this.getList()

})

}

}

}

</script>

<script>

export default {

created() {

const { params, query } = this.$route

const { path } = params

this.$router.replace({ path: '/' + path, query })

},

render: function(h) {

return h() // avoid warning message

}

}

</script>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleCreate">

新增

</el-button>

<!-- <el-select v-model="filterIsDel" style="width: 140px" class="filter-item" @change="handleFilter">-->

<!-- <el-option v-for="item in filterMap" :key="item.value" :label="item.label" :value="item.value" />-->

<!-- </el-select>-->

</div>

<el-table

:key="tableKey"

v-loading="listLoading"

:data="list"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<!-- <span>{{ row.updateAt | parseTime('{y}-{m}-{d} {h}:{i}') }}</span>-->

<span>{{ row.updateAt }}</span>

</template>

</el-table-column>

<el-table-column label="轮播图标题" min-width="120px" align="center">

<template slot-scope="{row}">

<span v-if="row.swiperStatus==='reviewed' || row.swiperStatus==='reviewing'">{{ row.swiperName }}</span>

<span v-else class="link-type" @click="handleUpdate(row)">{{ row.swiperName }}</span>

</template>

</el-table-column>

<el-table-column label="审核人" width="100px" align="center">

<template slot-scope="{row}">

<span style="color:orange;">{{ row.swiperReviewer }}</span>

</template>

</el-table-column>

<el-table-column label="状态" class-name="status-col" width="100px">

<template slot-scope="{row}">

<el-tag :type="row.swiperStatus | statusFilter">

{{ row.swiperStatus | statusNameFilter}}

</el-tag>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-button v-if="row.swiperStatus!=='reviewed' && row.swiperStatus!=='reviewing'" type="primary" size="mini" @click="handleUpdate(row)">

编辑

</el-button>

<el-button v-if="row.swiperStatus!=='reviewed' && row.swiperStatus!=='reviewing'" size="mini" type="success" @click="handleModifyStatus(row,'reviewing')">

提审

</el-button>

<el-button v-if="row.swiperStatus!=='draft' && row.swiperStatus!=='reviewed'" size="mini" @click="handleModifyStatus(row,'draft')">

撤回

</el-button>

<el-button v-if="row.swiperStatus!=='reviewed' && row.swiperStatus!=='reviewing'" size="mini" type="danger" @click="confirmDelete(row)">

删除

</el-button>

<el-tag v-if="row.swiperStatus==='reviewed'">

已通过审核，当前权限无法进行其他操作

</el-tag>

</template>

</el-table-column>

</el-table>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<el-dialog :title="textMap" :visible.sync="dialogFormVisible">

<el-form ref="dataForm" :rules="rules" :model="temp" label-position="left" label-width="100px" style="width: 400px; margin-left:50px;">

<el-form-item label="轮播图标题" prop="swiperName">

<el-input v-model="temp.swiperName" />

</el-form-item>

<!-- <el-form-item label="轮播图">-->

<!-- <el-upload-->

<!-- :action="this.$store.state.settings.uploadUrl"-->

<!-- :show-file-list="false"-->

<!-- :on-success="answerPicImageSuccess"-->

<!-- :before-upload="beforeUpload"-->

<!-- drag-->

<!-- prop="pic"-->

<!-- accept="image/png,image/gif,image/jpg,image/jpeg"-->

<!-- class="upload-demo" style="margin-top: 5%">-->

<!-- <img v-if="answerPicImageUrl" :src="answerPicImageUrl">-->

<!-- <i class="el-icon-upload"></i>-->

<!-- <div class="el-upload\_\_text">将文件拖到此处，或<em>点击上传</em></div>-->

<!-- </el-upload>-->

<!-- </el-form-item>-->

<el-form-item label="轮播图" prop="swiperPic">

<el-button v-show="!temp.swiperPic" type="primary" size="mini" @click="imageCropperShow()">

上传图片

</el-button>

<el-button v-show="temp.swiperPic" type="primary" size="mini" @click="imageCropperShow()">

更新图片

</el-button>

<el-button v-show="temp.swiperPic" type="primary" size="mini" @click="picShow(temp.swiperPic)">

预览

</el-button>

</el-form-item>

<el-form-item label="轮播图状态" prop="swiperStatus">

<el-select v-model="temp.swiperStatus" class="filter-item" placeholder="保存草稿/提交审核">

<el-option v-for="item in statusOptions" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer">

<el-button @click="dialogFormVisible = false">

取消

</el-button>

<el-button type="primary" @click="dialogStatus==='create'?createData():updateData()">

确认

</el-button>

</div>

</el-dialog>

<el-dialog title="提示" :visible.sync="delVisible" width="300px" center>

<div class="del-dialog-cnt">删除不可恢复，是否确定删除？</div>

<span slot="footer" class="dialog-footer">

<el-button @click="delVisible = false">取 消</el-button>

<el-button type="primary" @click="handleDelete(row)" >确 定</el-button>

</span>

</el-dialog>

<my-upload

method="POST"

field="file"

v-model="cropperShow"

:width=1920

:height=450

:url="this.$store.state.settings.uploadUrl"

lang-type='zh'

img-format='jpg'

img-bgc='#FFF'

:no-circle=true

@crop-upload-success="cropUploadSuccess">

</my-upload>

<el-dialog title="图片预览" :visible.sync="picVisible" width="800px" center>

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

<img :src="answerPicImageUrl" alt="" class="title-pic">

</div>

<span slot="footer" class="dialog-footer">

<el-button @click="picVisible = false">确定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import WebSwiperApi from "@/api/website/WebSwiperApi";

import 'babel-polyfill'; // es6 shim

import myUpload from 'vue-image-crop-upload';

export default {

name: 'swiper',

components: { Pagination, myUpload },

directives: { waves },

filters: {

statusNameFilter(status) {

const statusMap = {

reviewing: '审核中',

draft: '草稿',

reviewed: '已审核'

}

return statusMap

},

statusFilter(status) {

const statusMap = {

reviewing: 'warning',

reviewed: 'success',

draft: 'info',

}

return statusMap

}

},

data() {

return {

cropperShow: false,

picVisible: false,

filterIsDel: 1,

delVisible: false,

tableKey: 0,

list: null,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC'

},

// 已过审的不允许删除和存为草稿和提交审核

statusOptions: [{

label: '提交审核',

value: 'reviewing'

}, {

label: '存为草稿',

value: 'draft'

}],

showReviewer: false,

temp: {

swiperName: "",

swiperStatus: '',

swiperPic: '',

updateAt:''

},

dialogFormVisible: false,

dialogStatus: '',

textMap: {

update: '编辑',

create: '创建'

},

filterMap: [{

label: '全部',

value: 1

}, {

label: '未删除',

value: 2

}, {

label: '已删除',

value: 3

}],

rules: {

swiperName: ,

swiperStatus: ,

swiperPic: ,

},

downloadLoading: false,

answerPicImageUrl: ''

}

},

created() {

this.getList()

},

methods: {

// 刷新界面

getList() {

this.listLoading = true

if(this.filterIsDel === 3){

WebSwiperApi.findAllDel(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

}

else if(this.filterIsDel === 2){

WebSwiperApi.findAllExist(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

}

else{

WebSwiperApi.findAllByKeywords(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

}

this.listLoading = false

},

handleFilter() {

this.listQuery.page = 1

this.getList()

},

handleModifyStatus(row, status) {

this.listLoading = true

row.swiperStatus = status

this.temp = Object.assign({}, row) // copy obj

WebSwiperApi.update(this.temp)

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.filterIsDel = 1

this.handleFilter()

},

handleCreate() {

Object.keys(this.temp).forEach(key => (this.temp = ''))

this.dialogStatus = 'create'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

createData() {

this.$refs.validate((valid) => {

if (valid) {

WebSwiperApi.add(this.temp).then(() => {

this.list.unshift(this.temp)

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

})

}

})

},

handleUpdate(row) {

this.temp = Object.assign({}, row) // copy obj

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + this.temp.swiperPic

this.dialogStatus = 'update'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

updateData() {

this.$refs.validate((valid) => {

if (valid) {

const tempData = Object.assign({}, this.temp)

WebSwiperApi.update(tempData).then(() => {

for (const v of this.list) {

if (v.id === this.temp.id) {

const index = this.list.indexOf(v)

this.list.splice(index, 1, this.temp)

break

}

}

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

})

}

})

},

confirmDelete(row) {

// this.delVisible = true

if(confirm('确定要删除吗')===true){

this.handleDelete(row)

}

},

handleDelete(row) {

WebSwiperApi.delOne(row.id).then(data => {

this.getList()

})

this.$notify({

title: 'Success',

message: '成功删除',

type: 'success',

duration: 2000

})

const index = this.list.indexOf(row)

this.list.splice(index, 1)

},

answerPicImageSuccess (res, file) {

this.temp.swiperPic = res.data

this.answerPicImageUrl = URL.createObjectURL(file.raw)

},

beforeUpload (file) {

// 只允许上传8M以内大小的图片

const isLt8M = file.size / 1024 / 1024 < 8;

if (!isLt8M) {

this.$message.error('上传头像图片大小不能超过8MB!');

}

return isLt8M;

},

picShow(pic) {

this.picVisible = !this.picVisible

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + pic

},

// 显示图片上传模块

imageCropperShow() {

this.cropperShow = !this.cropperShow

},

// 图片上传成功后执行

cropUploadSuccess(jsonData, field){

this.temp.swiperPic = jsonData.data

this.$notify({

title: 'success',

message: '图片上传成功',

type: 'success',

duration: 2000

})

},

}

}

</script>

<style>

.title-pic{

width: 90%;

}

</style>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<el-button class="filter-item" style="margin-left: 10px;" type="primary" icon="el-icon-edit" @click="handleCreate">

新增

</el-button>

<!-- <el-select v-model="filterIsDel" style="width: 140px" class="filter-item" @change="handleFilter">-->

<!-- <el-option v-for="item in filterMap" :key="item.value" :label="item.label" :value="item.value" />-->

<!-- </el-select>-->

</div>

<el-table

:key="tableKey"

v-loading="listLoading"

:data="list"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<!-- <span>{{ row.updateAt | parseTime('{y}-{m}-{d} {h}:{i}') }}</span>-->

<span>{{ row.updateAt }}</span>

</template>

</el-table-column>

<el-table-column label="视频标题" min-width="120px" align="center">

<template slot-scope="{row}">

<span v-if="row.videoStatus==='reviewed' || row.videoStatus==='reviewing'">{{ row.videoTitle }}</span>

<span v-else class="link-type" @click="handleUpdate(row)">{{ row.videoTitle }}</span>

</template>

</el-table-column>

<el-table-column label="创作团队" width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoAuthor }}</span>

</template>

</el-table-column>

<el-table-column label="视频链接" width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoUrl }}</span>

</template>

</el-table-column>

<el-table-column label="视频简介" max-width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.videoProfile }}</span>

</template>

</el-table-column>

<el-table-column label="审核人" width="100px" align="center">

<template slot-scope="{row}">

<span style="color:orange;">{{ row.videoReviewer }}</span>

</template>

</el-table-column>

<el-table-column label="状态" class-name="status-col" width="100px">

<template slot-scope="{row}">

<el-tag :type="row.videoStatus | statusFilter">

{{ row.videoStatus | statusNameFilter}}

</el-tag>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-button v-if="row.videoStatus!=='reviewed' && row.videoStatus!=='reviewing'" type="primary" size="mini" @click="handleUpdate(row)">

编辑

</el-button>

<el-button v-if="row.videoStatus!=='reviewed' && row.videoStatus!=='reviewing' && row.videoStatus!=='reject'" size="mini" type="success" @click="handleModifyStatus(row,'reviewing')">

提审

</el-button>

<el-button v-if="row.videoStatus!=='draft' && row.videoStatus!=='reviewed' && row.videoStatus!=='reject'" size="mini" @click="handleModifyStatus(row,'draft')">

撤回

</el-button>

<el-button v-if="row.videoStatus!=='reviewed' && row.videoStatus!=='reviewing' && row.videoStatus!=='reject'" size="mini" type="danger" @click="confirmDelete(row)">

删除

</el-button>

<el-tag v-if="row.videoStatus==='reviewed'">

已通过审核，当前权限无法进行其他操作

</el-tag>

</template>

</el-table-column>

</el-table>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<el-dialog :title="textMap" :visible.sync="dialogFormVisible">

<el-form ref="dataForm" :rules="rules" :model="temp" label-position="left" label-width="100px" style="width: 400px; margin-left:50px;">

<el-form-item label="视频标题" prop="videoTitle">

<el-input v-model="temp.videoTitle" />

</el-form-item>

<el-form-item label="创作团队" prop="author">

<el-input v-model="temp.videoAuthor" />

</el-form-item>

<el-form-item label="视频简介" prop="profile">

<el-input v-model="temp.videoProfile" :autosize="{ minRows: 2, maxRows: 4}" type="textarea" placeholder="请输入简介" />

</el-form-item>

<el-form-item label="视频链接" prop="url">

<el-input v-model="temp.videoUrl" />

</el-form-item>

<!-- 以下为旧版封面上传方式-->

<!-- <el-form-item label="视频封面">-->

<!-- <el-upload-->

<!-- :action="this.$store.state.settings.uploadUrl"-->

<!-- :show-file-list="false"-->

<!-- :on-success="answerPicImageSuccess"-->

<!-- :before-upload="beforeUpload"-->

<!-- drag-->

<!-- prop="pic"-->

<!-- accept="image/png,image/gif,image/jpg,image/jpeg"-->

<!-- style="margin-top: 5%;">-->

<!-- <img v-if="answerPicImageUrl" :src="answerPicImageUrl" alt="" class="video-pic-image">-->

<!-- <i class="el-icon-upload"></i>-->

<!-- <div class="el-upload\_\_text">将文件拖到此处，或<em>点击上传</em></div>-->

<!-- </el-upload>-->

<!-- </el-form-item>-->

<el-form-item label="视频封面" prop="videoPic">

<el-button v-show="!temp.videoPic" type="primary" size="mini" @click="imageCropperShow()">

上传图片

</el-button>

<el-button v-show="temp.videoPic" type="primary" size="mini" @click="imageCropperShow()">

更新图片

</el-button>

<el-button v-show="temp.videoPic" type="primary" size="mini" @click="picShow(temp.videoPic)">

预览

</el-button>

</el-form-item>

<el-form-item label="视频状态" prop="videoStatus">

<el-select v-model="temp.videoStatus" class="filter-item" placeholder="保存草稿/提交审核">

<el-option v-for="item in statusOptions" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer">

<el-button @click="dialogFormVisible = false">

取消

</el-button>

<el-button type="primary" @click="dialogStatus==='create'?createData():updateData()">

确认

</el-button>

</div>

</el-dialog>

<el-dialog title="提示" :visible.sync="delVisible" width="300px" center>

<div class="del-dialog-cnt">删除不可恢复，是否确定删除？</div>

<span slot="footer" class="dialog-footer">

<el-button @click="delVisible = false">取 消</el-button>

<el-button type="primary" @click="handleDelete(row)" >确 定</el-button>

</span>

</el-dialog>

<my-upload

method="POST"

field="file"

v-model="cropperShow"

:width=600

:height=800

:url="this.$store.state.settings.uploadUrl"

lang-type='zh'

img-format='jpg'

img-bgc='#FFF'

:no-circle=true

@crop-upload-success="cropUploadSuccess">

</my-upload>

<el-dialog title="图片预览" :visible.sync="picVisible" width="400px" center>

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

<img :src="answerPicImageUrl" alt="" class="title-pic">

</div>

<span slot="footer" class="dialog-footer">

<el-button @click="picVisible = false">确定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import WebVideoApi from "@/api/website/WebVideoApi";

import 'babel-polyfill'; // es6 shim

import myUpload from 'vue-image-crop-upload';

export default {

name: 'websiteVideo',

components: { Pagination, myUpload },

directives: { waves },

filters: {

statusNameFilter(status) {

const statusMap = {

reviewing: '审核中',

draft: '草稿',

reviewed: '已审核',

reject: '被驳回'

}

return statusMap

},

statusFilter(status) {

const statusMap = {

reviewing: 'warning',

reviewed: 'success',

draft: 'info',

reject: 'danger'

}

return statusMap

}

},

data() {

return {

picVisible: false,

cropperShow: false,

filterIsDel: 1,

delVisible: false,

tableKey: 0,

list: null,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC'

},

// 已过审的不允许删除和存为草稿和提交审核

statusOptions: [{

label: '提交审核',

value: 'reviewing'

}, {

label: '存为草稿',

value: 'draft'

}],

showReviewer: false,

temp: {

videoAuthor: "",

videoProfile: '',

videoTitle: '',

videoStatus: '',

videoUrl: '',

videoReviewer: '',

videoPic: '',

updateAt:''

},

dialogFormVisible: false,

dialogStatus: '',

textMap: {

update: '编辑',

create: '创建'

},

filterMap: [{

label: '全部',

value: 1

}, {

label: '未删除',

value: 2

}, {

label: '已删除',

value: 3

}],

rules: {

videoTitle: ,

videoStatus: ,

videoPic: ,

videoProfile: ,

},

downloadLoading: false,

answerPicImageUrl: ''

}

},

created() {

this.getList()

},

methods: {

// 刷新界面

getList() {

this.listLoading = true

WebVideoApi.findAllByKeywords(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

this.listLoading = false

},

handleFilter() {

this.listQuery.page = 1

this.getList()

},

handleModifyStatus(row, status) {

this.listLoading = true

row.videoStatus = status

this.temp = Object.assign({}, row) // copy obj

WebVideoApi.update(this.temp).then(() => {

for (const v of this.list) {

if (v.id === this.temp.id) {

const index = this.list.indexOf(v)

this.list.splice(index, 1, this.temp)

break

}

}

})

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

sortChange(data) {

const { prop, order } = data

if (prop === 'date') {

this.sortByID(order)

}

},

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

this.filterIsDel = 1

this.handleFilter()

},

handleCreate() {

Object.keys(this.temp).forEach(key => (this.temp = ''))

this.dialogStatus = 'create'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

createData() {

this.$refs.validate((valid) => {

if (valid) {

WebVideoApi.add(this.temp).then(() => {

this.list.unshift(this.temp)

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

})

}

})

},

handleUpdate(row) {

this.temp = Object.assign({}, row) // copy obj

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + this.temp.videoPic

this.dialogStatus = 'update'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

updateData() {

this.$refs.validate((valid) => {

if (valid) {

const tempData = Object.assign({}, this.temp)

WebVideoApi.update(tempData).then(() => {

for (const v of this.list) {

if (v.id === this.temp.id) {

const index = this.list.indexOf(v)

this.list.splice(index, 1, this.temp)

break

}

}

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

})

}

})

},

confirmDelete(row) {

// this.delVisible = true

if(confirm('确定要删除吗')===true){

this.handleDelete(row)

}

},

handleDelete(row) {

WebVideoApi.delOne(row.id).then(data => {

this.getList()

})

this.$notify({

title: 'Success',

message: '成功删除',

type: 'success',

duration: 2000

})

const index = this.list.indexOf(row)

this.list.splice(index, 1)

},

answerPicImageSuccess (res, file) {

this.temp.videoPic = res.data

this.answerPicImageUrl = URL.createObjectURL(file.raw)

},

beforeUpload (file) {

// 只允许上传8M以内大小的图片

const isLt8M = file.size / 1024 / 1024 < 8;

if (!isLt8M) {

this.$message.error('上传头像图片大小不能超过8MB!');

}

return isLt8M;

},

picShow(pic) {

this.picVisible = !this.picVisible

this.answerPicImageUrl = this.$store.state.settings.callbackUrl + pic

},

// 显示图片上传模块

imageCropperShow() {

this.cropperShow = !this.cropperShow

},

// 图片上传成功后执行

cropUploadSuccess(jsonData, field){

this.temp.videoPic = jsonData.data

this.$notify({

title: 'success',

message: '图片上传成功',

type: 'success',

duration: 2000

})

},

}

}

</script>

<style>

.title-pic{

width: 60%;

}

</style>

<template>

<div class="material-input\_\_component" :class="computedClasses">

<input

v-if="type === 'email'"

type="email"

class="material-input"

:name="name"

:id="id"

:placeholder="placeholder"

v-model="valueCopy"

:readonly="readonly"

:disabled="disabled"

:autocomplete="autocomplete"

:required="required"

@focus="handleFocus(true)"

@blur="handleFocus(false)"

@input="handleModelInput"

>

<input

v-if="type === 'url'"

type="url"

class="material-input"

:name="name"

:id="id"

:placeholder="placeholder"

v-model="valueCopy"

:readonly="readonly"

:disabled="disabled"

:autocomplete="autocomplete"

:required="required"

@focus="handleFocus(true)"

@blur="handleFocus(false)"

@input="handleModelInput"

>

<input

v-if="type === 'number'"

type="number"

class="material-input"

:name="name"

:id="id"

:placeholder="placeholder"

v-model="valueCopy"

:readonly="readonly"

:disabled="disabled"

:autocomplete="autocomplete"

:max="max"

:min="min"

:step="step"

:minlength="minlength"

:maxlength="maxlength"

:required="required"

@focus="handleFocus(true)"

@blur="handleFocus(false)"

@input="handleModelInput"

>

<input

v-if="type === 'password'"

type="password"

class="material-input"

:name="name"

:id="id"

:placeholder="placeholder"

v-model="valueCopy"

:readonly="readonly"

:disabled="disabled"

:autocomplete="autocomplete"

:max="max"

:min="min"

:step="step"

:required="required"

@focus="handleFocus(true)"

@blur="handleFocus(false)"

@input="handleModelInput"

>

<input

v-if="type === 'tel'"

type="tel"

class="material-input"

:name="name"

:id="id"

:placeholder="placeholder"

v-model="valueCopy"

:readonly="readonly"

:disabled="disabled"

:autocomplete="autocomplete"

:required="required"

@focus="handleFocus(true)"

@blur="handleFocus(false)"

@input="handleModelInput"

>

<input

v-if="type === 'text'"

type="text"

class="material-input"

:name="name"

:id="id"

:placeholder="placeholder"

v-model="valueCopy"

:readonly="readonly"

:disabled="disabled"

:autocomplete="autocomplete"

:minlength="minlength"

:maxlength="maxlength"

:required="required"

@focus="handleFocus(true)"

@blur="handleFocus(false)"

@input="handleModelInput"

>

<span class="material-input-bar"></span>

<label class="material-label">

<slot></slot>

</label>

<div v-if="errorMessages" class="material-errors">

<div v-for="error in computedErrors" class="material-error">

{{ error }}

</div>

</div>

</div>

</template>

<script>

export default {

name: 'material-input',

computed: {

computedErrors () {

return typeof this.errorMessages === 'string'

? : this.errorMessages

},

computedClasses () {

return {

'material--active': this.focus,

'material--disabled': this.disabled,

'material--has-errors': Boolean(

!this.valid ||

(this.errorMessages && this.errorMessages.length)),

'material--raised': Boolean(

this.focus ||

this.valueCopy || // has value

(this.placeholder && !this.valueCopy)) // has placeholder

}

}

},

data () {

return {

valueCopy: null,

focus: false,

valid: true

}

},

beforeMount () {

// Here we are following the Vue2 convention on custom v-model:

// https://github.com/vuejs/vue/issues/2873#issuecomment-223759341

this.copyValue(this.value)

},

methods: {

handleModelInput (event) {

this.$emit('input', event.target.value, event)

this.handleValidation()

},

handleFocus (focused) {

this.focus = focused

},

handleValidation () {

this.valid = this.$el ? this.$el.querySelector(

'.material-input').validity.valid : this.valid

},

copyValue (value) {

this.valueCopy = value

this.handleValidation()

}

},

watch: {

value (newValue) {

// This watch works from the code side of the 2-way-binding:

this.copyValue(newValue)

}

},

props: {

id: {

type: String,

default: null

},

name: {

type: String,

default: null

},

type: {

type: String,

default: 'text'

},

value: {

default: null

},

placeholder: {

type: String,

default: null

},

readonly: {

type: Boolean,

default: false

},

disabled: {

type: Boolean,

default: false

},

min: {

type: String,

default: null

},

max: {

type: String,

default: null

},

step: {

type: Number,

default: null

},

minlength: {

type: Number,

default: null

},

maxlength: {

type: Number,

default: null

},

required: {

type: Boolean,

default: true

},

autocomplete: {

type: String,

default: null

},

errorMessages: {

type: ,

default: null

}

}

}

</script>

<style lang="scss">

// Fonts:

$font-size-base: 16px;

$font-size-small: 14px;

$font-size-smallest: 12px;

$font-weight-normal: normal;

// Utils

$spacer: 10px;

$transition: 0.2s ease all;

// Base clases:

%base-bar-pseudo {

content: '';

height: 1px;

width: 0;

bottom: 0;

position: absolute;

transition: $transition

}

// Mixins:

@mixin slided-top() {

top: -2 \* $spacer;

font-size: $font-size-small;

}

// Component:

.material-input\_\_component {

margin-top: 45px;

position: relative;

\* {

box-sizing: border-box;

}

.material-input {

font-size: $font-size-base;

padding: $spacer $spacer $spacer $spacer / 2;

display: block;

width: 100%;

border: none;

border-radius: 0;

&:focus {

outline: none;

border: none;

border-bottom: 1px solid transparent; // fixes the height issue

}

}

.material-label {

font-size: $font-size-base;

font-weight: $font-weight-normal;

position: absolute;

pointer-events: none;

left: 0;

top: $spacer;

transition: $transition;

}

.material-input-bar {

position: relative;

display: block;

width: 100%;

&:before {

@extend %base-bar-pseudo;

left: 50%;

}

&:after {

@extend %base-bar-pseudo;

right: 50%;

}

}

// Disabled state:

&.material--disabled {

.material-input {

border-bottom-style: dashed;

}

}

// Raised state:

&.material--raised {

.material-label {

@include slided-top();

}

}

// Active state:

&.material--active {

.material-input-bar {

&:before,

&:after {

width: 50%;

}

}

}

// Errors:

.material-errors {

position: relative;

overflow: hidden;

.material-error {

font-size: $font-size-smallest;

line-height: $font-size-smallest + 2px;

overflow: hidden;

margin-top: 0;

padding-top: $spacer / 2;

padding-right: $spacer / 2;

padding-left: 0;

}

}

}

// Theme:

$color-white: white;

$color-grey: #9E9E9E;

$color-grey-light: #E0E0E0;

$color-blue: #2196F3;

$color-red: #F44336;

$color-black: black;

.material-input\_\_component {

background: $color-white;

.material-input {

background: none;

color: $color-black;

border-bottom: 1px solid $color-grey-light;

}

.material-label {

color: $color-grey;

}

.material-input-bar {

&:before,

&:after {

background: $color-blue;

}

}

// Active state:

&.material--active {

.material-label {

color: $color-blue;

}

}

// Errors:

&.material--has-errors {

// These styles are required

// for custom validation:

&.material--active .material-label {

color: $color-red;

}

.material-input-bar {

&:before,

&:after {

background: $color-red;

}

}

.material-errors {

color: $color-red;

}

}

}

</style>

<template>

<div :class="{'hidden':hidden}" class="pagination-container">

<el-pagination

:background="background"

:current-page.sync="currentPage"

:page-size.sync="pageSize"

:layout="layout"

:page-sizes="pageSizes"

:total="total"

v-bind="$attrs"

@size-change="handleSizeChange"

@current-change="handleCurrentChange"

/>

</div>

</template>

<script>

import { scrollTo } from '@/utils/scroll-to'

export default {

name: 'Pagination',

props: {

total: {

required: true,

type: Number

},

page: {

type: Number,

default: 1

},

limit: {

type: Number,

default: 20

},

pageSizes: {

type: Array,

default() {

return

}

},

layout: {

type: String,

default: 'total, sizes, prev, pager, next, jumper'

},

background: {

type: Boolean,

default: true

},

autoScroll: {

type: Boolean,

default: true

},

hidden: {

type: Boolean,

default: false

}

},

computed: {

currentPage: {

get() {

return this.page

},

set(val) {

this.$emit('update:page', val)

}

},

pageSize: {

get() {

return this.limit

},

set(val) {

this.$emit('update:limit', val)

}

}

},

methods: {

handleSizeChange(val) {

this.$emit('pagination', { page: this.currentPage, limit: val })

if (this.autoScroll) {

scrollTo(0, 800)

}

},

handleCurrentChange(val) {

this.$emit('pagination', { page: val, limit: this.pageSize })

if (this.autoScroll) {

scrollTo(0, 800)

}

}

}

}

</script>

<style scoped>

.pagination-container {

background: #fff;

padding: 32px 16px;

}

.pagination-container.hidden {

display: none;

}

</style>

<template>

<div :style="{height:height+'px',zIndex:zIndex}">

<div

:class="className"

:style="{top:(isSticky ? stickyTop +'px' : ''),zIndex:zIndex,position:position,width:width,height:height+'px'}"

>

<slot>

<div>sticky</div>

</slot>

</div>

</div>

</template>

<script>

export default {

name: 'Sticky',

props: {

stickyTop: {

type: Number,

default: 0

},

zIndex: {

type: Number,

default: 1

},

className: {

type: String,

default: ''

}

},

data() {

return {

active: false,

position: '',

width: undefined,

height: undefined,

isSticky: false

}

},

mounted() {

this.height = this.$el.getBoundingClientRect().height

window.addEventListener('scroll', this.handleScroll)

window.addEventListener('resize', this.handleResize)

},

activated() {

this.handleScroll()

},

destroyed() {

window.removeEventListener('scroll', this.handleScroll)

window.removeEventListener('resize', this.handleResize)

},

methods: {

sticky() {

if (this.active) {

return

}

this.position = 'fixed'

this.active = true

this.width = this.width + 'px'

this.isSticky = true

},

handleReset() {

if (!this.active) {

return

}

this.reset()

},

reset() {

this.position = ''

this.width = 'auto'

this.active = false

this.isSticky = false

},

handleScroll() {

const width = this.$el.getBoundingClientRect().width

this.width = width || 'auto'

const offsetTop = this.$el.getBoundingClientRect().top

if (offsetTop < this.stickyTop) {

this.sticky()

return

}

this.handleReset()

},

handleResize() {

if (this.isSticky) {

this.width = this.$el.getBoundingClientRect().width + 'px'

}

}

}

}

</script>

<template>

<div v-if="isExternal" :style="styleExternalIcon" class="svg-external-icon svg-icon" v-on="$listeners" />

<svg v-else :class="svgClass" aria-hidden="true" v-on="$listeners">

<use :href="iconName" />

</svg>

</template>

<script>

// doc: https://panjiachen.github.io/vue-element-admin-site/feature/component/svg-icon.html#usage

import { isExternal } from '@/utils/validate'

export default {

name: 'SvgIcon',

props: {

iconClass: {

type: String,

required: true

},

className: {

type: String,

default: ''

}

},

computed: {

isExternal() {

return isExternal(this.iconClass)

},

iconName() {

return `#icon-${this.iconClass}`

},

svgClass() {

if (this.className) {

return 'svg-icon ' + this.className

} else {

return 'svg-icon'

}

},

styleExternalIcon() {

return {

mask: `url(${this.iconClass}) no-repeat 50% 50%`,

'-webkit-mask': `url(${this.iconClass}) no-repeat 50% 50%`

}

}

}

}

</script>

<style scoped>

.svg-icon {

width: 1em;

height: 1em;

vertical-align: -0.15em;

fill: currentColor;

overflow: hidden;

}

.svg-external-icon {

background-color: currentColor;

mask-size: cover!important;

display: inline-block;

}

</style>

<template>

<div class="tinymce-editor">

<Editor

:id="editorId"

v-model="editorValue"

:init="editorInit"

:disabled="disabled"

@onClick="handleClick"

/>

</div>

</template>

<script>

// 引入组件

import tinymce from 'tinymce/tinymce'

import Editor from '@tinymce/tinymce-vue'

// 引入富文本编辑器主题的js和css

import 'tinymce/themes/silver/theme.min.js'

import 'tinymce/skins/ui/oxide/skin.min.css'

// 扩展插件

import 'tinymce/plugins/image'

import 'tinymce/plugins/link'

import 'tinymce/plugins/code'

import 'tinymce/plugins/table'

import 'tinymce/plugins/lists'

import 'tinymce/plugins/wordcount' // 字数统计插件

export default {

name: 'TinymceEditor',

components: { Editor },

props: {

id: {

type: String,

default: 'tinymceEditor'

},

value: {

type: String,

default: ''

},

disabled: {

type: Boolean,

default: false

},

plugins: {

type: ,

default: 'link lists image code table wordcount'

},

toolbar: {

type: ,

default: 'bold italic underline strikethrough | fontsizeselect | forecolor backcolor | alignleft aligncenter alignright alignjustify | bullist numlist | outdent indent blockquote | undo redo | link unlink image code | removeformat'

}

},

data() {

return {

editorInit: {

language\_url: './static/tinymce/langs/zh\_CN.js',

language: 'zh\_CN',

skin\_url: './static/tinymce/skins/ui/oxide',

height: 300,

plugins: this.plugins,

toolbar: this.toolbar,

statusbar: true, // 底部的状态栏

menubar: true, // 最上方的菜单

branding: false, // 水印“Powered by TinyMCE”

images\_upload\_handler: (blobInfo, success, failure) => {

this.$emit('handleImgUpload', blobInfo, success, failure)

}

},

editorId: this.id,

editorValue: this.value

}

},

watch: {

editorValue(newValue) {

this.$emit('input', newValue)

}

},

mounted() {

tinymce.init({})

},

methods: {

// https://github.com/tinymce/tinymce-vue => All available events

handleClick(e) {

this.$emit('onClick', e, tinymce)

},

clear() {

this.editorValue = ''

},

handleImgUpload(blobInfo, success, failure) {

}

}

}

</script>

<template>

<div :class="{'has-logo':showLogo}">

<logo v-if="showLogo" :collapse="isCollapse" />

<el-scrollbar wrap-class="scrollbar-wrapper">

<el-menu

:default-active="activeMenu"

:collapse="isCollapse"

:background-color="variables.menuBg"

:text-color="variables.menuText"

:unique-opened="false"

:active-text-color="variables.menuActiveText"

:collapse-transition="false"

mode="vertical"

>

<sidebar-item v-for="route in permission\_routes" :key="route.path" :item="route" :base-path="route.path" />

</el-menu>

</el-scrollbar>

</div>

</template>

<script>

import { mapGetters } from 'vuex'

import Logo from './Logo'

import SidebarItem from './SidebarItem'

import variables from '@/styles/variables.scss'

export default {

components: { SidebarItem, Logo },

computed: {

...mapGetters([

'permission\_routes',

'sidebar'

]),

activeMenu() {

const route = this.$route

const { meta, path } = route

// if set path, the sidebar will highlight the path you set

if (meta.activeMenu) {

return meta.activeMenu

}

return path

},

showLogo() {

return this.$store.state.settings.sidebarLogo

},

variables() {

return variables

},

isCollapse() {

return !this.sidebar.opened

}

}

}

</script>

<template>

<div :class="classObj" class="app-wrapper">

<div v-if="device==='mobile'&&sidebar.opened" class="drawer-bg" @click="handleClickOutside" />

<sidebar class="sidebar-container" />

<div class="main-container">

<div :class="{'fixed-header':fixedHeader}">

<navbar />

</div>

<app-main />

</div>

</div>

</template>

<script>

import { Navbar, Sidebar, AppMain } from './components'

import ResizeMixin from './mixin/ResizeHandler'

export default {

name: 'Layout',

components: {

Navbar,

Sidebar,

AppMain

},

mixins: ,

computed: {

sidebar() {

return this.$store.state.app.sidebar

},

device() {

return this.$store.state.app.device

},

fixedHeader() {

return this.$store.state.settings.fixedHeader

},

classObj() {

return {

hideSidebar: !this.sidebar.opened,

openSidebar: this.sidebar.opened,

withoutAnimation: this.sidebar.withoutAnimation,

mobile: this.device === 'mobile'

}

}

},

methods: {

handleClickOutside() {

this.$store.dispatch('app/closeSideBar', { withoutAnimation: false })

}

}

}

</script>

<style lang="scss" scoped>

@import "~@/styles/mixin.scss";

@import "~@/styles/variables.scss";

.app-wrapper {

@include clearfix;

position: relative;

height: 100%;

width: 100%;

&.mobile.openSidebar{

position: fixed;

top: 0;

}

}

.drawer-bg {

background: #000;

opacity: 0.3;

width: 100%;

top: 0;

height: 100%;

position: absolute;

z-index: 999;

}

.fixed-header {

position: fixed;

top: 0;

right: 0;

z-index: 9;

width: calc(100% - #{$sideBarWidth});

transition: width 0.28s;

}

.hideSidebar .fixed-header {

width: calc(100% - 54px)

}

.mobile .fixed-header {

width: 100%;

}

</style>

<template>

<el-breadcrumb class="app-breadcrumb" separator="/">

<transition-group name="breadcrumb">

<el-breadcrumb-item v-for="(item,index) in levelList" :key="item.path">

<span v-if="item.redirect==='noRedirect'||index==levelList.length-1" class="no-redirect">{{ item.meta.title }}</span>

<a v-else @click.prevent="handleLink(item)">{{ item.meta.title }}</a>

</el-breadcrumb-item>

</transition-group>

</el-breadcrumb>

</template>

<script>

import \* as pathToRegexp from 'path-to-regexp'

export default {

data() {

return {

levelList: null

}

},

watch: {

$route() {

this.getBreadcrumb()

}

},

created() {

this.getBreadcrumb()

},

methods: {

getBreadcrumb() {

// only show routes with meta.title

let matched = this.$route.matched.filter(item => item.meta && item.meta.title)

const first = matched

if (!this.isDashboard(first)) {

matched = .concat(matched)

}

this.levelList = matched.filter(item => item.meta && item.meta.title && item.meta.breadcrumb !== false)

},

isDashboard(route) {

const name = route && route.name

if (!name) {

return false

}

return name.trim().toLocaleLowerCase() === 'Dashboard'.toLocaleLowerCase()

},

pathCompile(path) {

// To solve this problem https://github.com/PanJiaChen/vue-element-admin/issues/561

const { params } = this.$route

var toPath = pathToRegexp.compile(path)

return toPath(params)

},

handleLink(item) {

const { redirect, path } = item

if (redirect) {

this.$router.push(redirect)

return

}

this.$router.push(this.pathCompile(path))

}

}

}

</script>

<style lang="scss" scoped>

.app-breadcrumb.el-breadcrumb {

display: inline-block;

font-size: 14px;

line-height: 50px;

margin-left: 8px;

.no-redirect {

color: #97a8be;

cursor: text;

}

}

</style>

<script>

export default {

name: 'MenuItem',

functional: true,

props: {

icon: {

type: String,

default: ''

},

title: {

type: String,

default: ''

}

},

render(h, context) {

const { icon, title } = context.props

const vnodes =

if (icon) {

vnodes.push(<svg-icon icon-class={icon}/>)

}

if (title) {

vnodes.push(<span slot='title'>{(title)}</span>)

}

return vnodes

}

}

</script>

<template>

<!-- eslint-disable vue/require-component-is -->

<component v-bind="linkProps(to)">

<slot />

</component>

</template>

<script>

import { isExternal } from '@/utils/validate'

export default {

props: {

to: {

type: String,

required: true

}

},

methods: {

linkProps(url) {

if (isExternal(url)) {

return {

is: 'a',

href: url,

target: '\_blank',

rel: 'noopener'

}

}

return {

is: 'router-link',

to: url

}

}

}

}

</script>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<router-link style="margin-left: 10px;" :to="'/web\_management/article/create'" class="link-type">

<el-button class="filter-item" type="primary" icon="el-icon-edit">

新增

</el-button>

</router-link>

</div>

<el-table

:key="tableKey"

v-loading="listLoading"

:data="list"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.updateAt }}</span>

</template>

</el-table-column>

<el-table-column label="标题" min-width="150px" align="center">

<template slot-scope="{row}">

<span v-if="row.status==='reviewed' || row.status==='reviewing'">{{ row.title }}</span>

<router-link v-else :to="'/web\_management/article/edit/'+row.id" class="link-type">

<span>{{ row.title }}</span>

</router-link>

</template>

</el-table-column>

<el-table-column label="作者" width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.author }}</span>

</template>

</el-table-column>

<el-table-column label="审核人" width="100px" align="center">

<template slot-scope="{row}">

<span style="color:orange;">{{ row.reviewer }}</span>

</template>

</el-table-column>

<el-table-column label="阅读量" align="center">

<template slot-scope="{row}">

<span>{{ row.readings }}</span>

</template>

</el-table-column>

<el-table-column label="状态" class-name="status-col" width="100px">

<template slot-scope="{row}">

<el-tag :type="row.status | statusFilter">

{{ row.status | statusNameFilter}}

</el-tag>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<router-link style="margin-right: 10px;" v-if="row.status!=='reviewed' && row.status!=='reviewing'" :to="'/web\_management/article/edit/'+row.id">

<el-button type="primary" size="mini">

编辑

</el-button>

</router-link>

<el-button v-if="row.status!=='reviewed' && row.status!=='reviewing' && row.status!=='reject'" size="mini" type="success" @click="handleModifyStatus(row,'reviewing')">

提审

</el-button>

<el-button v-if="row.status!=='draft' && row.status!=='reviewed' && row.status!=='reject'" size="mini" @click="handleModifyStatus(row,'draft')">

撤回

</el-button>

<el-button v-if="row.status!=='reviewed' && row.status!=='reviewing' && row.status!=='reject'" size="mini" type="danger" @click="confirmDelete(row)">

删除

</el-button>

<el-tag v-if="row.status==='reviewed'">

已通过审核，当前权限无法进行其他操作

</el-tag>

</template>

</el-table-column>

</el-table>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<el-dialog :title="textMap" :visible.sync="dialogFormVisible">

<el-form ref="dataForm" :rules="rules" :model="temp" label-position="left" label-width="100px" style="width: 400px; margin-left:50px;">

<el-form-item label="文章标题" prop="title">

<el-input v-model="temp.title" />

</el-form-item>

<el-form-item label="文章作者" prop="author">

<el-input v-model="temp.author" />

</el-form-item>

<el-form-item label="文章正文" prop="article">

<el-input v-model="temp.article" />

</el-form-item>

<el-form-item label="文章状态" prop="status">

<el-select v-model="temp.status" class="filter-item" placeholder="保存草稿/提交审核">

<el-option v-for="item in statusOptions" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer">

<el-button @click="dialogFormVisible = false">

取消

</el-button>

<el-button type="primary" @click="dialogStatus==='create'?createData():updateData()">

确认

</el-button>

</div>

</el-dialog>

<el-dialog title="提示" :visible.sync="delVisible" width="300px" center>

<div class="del-dialog-cnt">删除不可恢复，是否确定删除？</div>

<span slot="footer" class="dialog-footer">

<el-button @click="delVisible = false">取 消</el-button>

<el-button type="primary" @click="handleDelete(row)" >确 定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import WebArticleApi from "@/api/website/WebArticleApi";

export default {

name: 'List',

components: { Pagination },

directives: { waves },

filters: {

statusNameFilter(status) {

const statusMap = {

reviewing: '审核中',

draft: '草稿',

reviewed: '已审核',

reject: '被驳回'

}

return statusMap

},

statusFilter(status) {

const statusMap = {

reviewed: 'success',

draft: 'info',

reviewing: 'warning',

reject: 'danger'

}

return statusMap

}

},

data() {

return {

delVisible: false,

tableKey: 0,

list: null,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC'

},

// 已过审的不允许删除和存为草稿和提交审核

statusOptions: [{

label: '提交审核',

value: 'reviewing'

}, {

label: '存为草稿',

value: 'draft'

}],

showReviewer: false,

temp: {

author: "",

article: ``,

title: '',

status: '',

updateAt: '',

},

dialogFormVisible: false,

dialogStatus: '',

textMap: {

update: '编辑',

create: '创建'

},

rules: {

status: ,

title:

},

downloadLoading: false

}

},

created() {

this.getList()

},

methods: {

// 刷新界面

getList() {

this.listLoading = true

WebArticleApi.findAllByKeywords(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

this.listLoading = false

},

handleFilter() {

this.listQuery.page = 1

this.getList()

},

handleModifyStatus(row, status) {

this.listLoading = true

row.status = status

this.temp = Object.assign({}, row) // copy obj

WebArticleApi.update(this.temp)

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

sortChange(data) {

const { prop, order } = data

console.log(order)

console.log(prop)

if (prop === 'date') {

this.sortByID(order)

}

},

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

console.log(this.listQuery.direction)

this.handleFilter()

},

handleCreate() {

Object.keys(this.temp).forEach(key => (this.temp = ''))

this.dialogStatus = 'create'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

createData() {

this.$refs.validate((valid) => {

if (valid) {

WebArticleApi.add(this.temp).then(data => {

console.log(data)

this.list.unshift(this.temp)

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

})

}

})

},

handleUpdate(row) {

this.temp = Object.assign({}, row) // copy obj

this.dialogStatus = 'update'

this.dialogFormVisible = true

this.$nextTick(() => {

this.$refs.clearValidate()

})

},

updateData() {

this.$refs.validate((valid) => {

if (valid) {

const tempData = Object.assign({}, this.temp)

WebArticleApi.update(tempData).then(() => {

this.getList()

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '更新成功',

type: 'success',

duration: 2000

})

})

}

})

},

confirmDelete(row) {

// this.delVisible = true

if(confirm('确定要删除吗')===true){

this.handleDelete(row)

}

},

handleDelete(row) {

WebArticleApi.delOne(row.id).then(data => {

console.log(data)

this.getList()

})

this.$notify({

title: 'Success',

message: '成功删除',

type: 'success',

duration: 2000

})

}

}

}

</script>

<template>

<div class="app-container">

<div class="filter-container">

<el-input v-model="listQuery.keywords" placeholder="请输入搜索内容" style="width: 200px;padding-right: 10px;" class="filter-item" @keyup.enter.native="handleFilter" />

<el-button v-waves class="filter-item" type="primary" icon="el-icon-search" @click="handleFilter">

搜索

</el-button>

<router-link style="margin-left: 10px;" :to="'/admin\_management/verify/article/create'" class="link-type">

<el-button class="filter-item" type="primary" icon="el-icon-edit">

新增

</el-button>

</router-link>

<el-select v-model="filterIsDel" style="width: 140px" class="filter-item" @change="handleFilter">

<el-option v-for="item in filterMap" :key="item.value" :label="item.label" :value="item.value" />

</el-select>

</div>

<el-table

:key="tableKey"

v-loading="listLoading"

:data="list"

border

fit

highlight-current-row

style="width: 100%;"

@sort-change="sortChange"

>

<el-table-column label="发布日期" prop="date" sortable="custom" width="200px" align="center">

<template slot-scope="{row}">

<span>{{ row.createAt }}</span>

</template>

</el-table-column>

<el-table-column label="标题" min-width="150px" align="center">

<template slot-scope="{row}">

<span>{{ row.title }}</span>

</template>

</el-table-column>

<el-table-column label="作者" width="100px" align="center">

<template slot-scope="{row}">

<span>{{ row.author }}</span>

</template>

</el-table-column>

<el-table-column label="阅读量" align="center">

<template slot-scope="{row}">

<span>{{ row.readings }}</span>

</template>

</el-table-column>

<el-table-column label="状态" class-name="status-col" width="100px">

<template slot-scope="{row}">

<el-tag :type="row.status | statusFilter">

{{ row.status | statusNameFilter}}

</el-tag>

</template>

</el-table-column>

<el-table-column label="上架/下架" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<el-button v-if="row.isDel===true" size="mini" type="success" @click="handleModifyIsDel(row, false)">

上架

</el-button>

<el-button v-if="row.isDel===false" size="mini" @click="handleModifyIsDel(row, true)">

下架

</el-button>

</template>

</el-table-column>

<el-table-column label="操作" align="center" width="300px" class-name="small-padding fixed-width">

<template slot-scope="{row}">

<router-link style="margin-right: 10px;" v-if="row.status==='reviewing'" :to="'/admin\_management/verify/article/edit/'+row.id">

<el-button type="primary" size="mini">

审阅

</el-button>

</router-link>

<router-link style="margin-right: 10px;" v-if="row.status==='reviewed'" :to="'/admin\_management/verify/article/edit/'+row.id">

<el-button type="primary" size="mini">

修改

</el-button>

</router-link>

<el-button v-if="row.status==='reject'" size="mini" type="success" @click="handleModifyStatus(row,'reviewing')">

待定

</el-button>

<el-button v-if="row.status==='reviewed'" size="mini" @click="handleModifyStatus(row,'reject')">

驳回

</el-button>

</template>

</el-table-column>

</el-table>

<pagination v-show="total>0" :total="total" :page.sync="listQuery.page" :size.sync="listQuery.size" @pagination="getList" />

<el-dialog title="提示" :visible.sync="delVisible" width="300px" center>

<div class="del-dialog-cnt">删除不可恢复，是否确定删除？</div>

<span slot="footer" class="dialog-footer">

<el-button @click="delVisible = false">取 消</el-button>

<el-button type="primary" @click="handleDelete(row)" >确 定</el-button>

</span>

</el-dialog>

</div>

</template>

<script>

import waves from '@/directive/waves' // waves directive

import Pagination from '@/components/Pagination' // secondary package based on el-pagination

import WebVerifyApi from "@/api/admin/WebVerifyApi";

import WebArticleApi from "@/api/website/WebArticleApi";

export default {

name: 'List',

components: { Pagination },

directives: { waves },

filters: {

statusNameFilter(status) {

const statusMap = {

reviewing: '审核中',

draft: '草稿',

reviewed: '已审核',

reject: '驳回'

}

return statusMap

},

statusFilter(status) {

const statusMap = {

reviewed: 'success',

draft: 'info',

reviewing: 'warning',

reject: 'danger'

}

return statusMap

}

},

data() {

return {

delVisible: false,

tableKey: 0,

filterIsDel: 1,

list: null,

total: 0,

listLoading: true,

listQuery: {

page: 1,

size: 20,

keywords: "",

direction: 'DESC',

active: ''

},

// 已过审的不允许删除和存为草稿和提交审核

statusOptions: [{

label: '提交审核',

value: 'reviewing'

}, {

label: '存为草稿',

value: 'draft'

}],

showReviewer: false,

temp: {

readings: '',

author: "",

article: ``,

title: '',

status: '',

createAt: '',

isDel:''

},

dialogFormVisible: false,

dialogStatus: '',

textMap: {

update: '编辑',

create: '创建'

},

filterMap: [

{ label: '全部', value: 1 },

{ label: '未下架', value: 'false' },

{ label: '已下架', value: 'true'}

],

rules: {

status: ,

title:

},

downloadLoading: false

}

},

created() {

this.getList()

},

methods: {

// 刷新界面

getList() {

this.listLoading = true

if(this.filterIsDel === 1){

WebVerifyApi.findAllArticle(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

}

else{

this.listQuery.active = this.filterIsDel

WebVerifyApi.findClassArticle(this.listQuery).then(data => {

this.list = data.data

this.total = data.total;

})

}

this.listLoading = false

},

handleFilter() {

this.listQuery.page = 1

this.getList()

},

handleModifyStatus(row, status) {

this.listLoading = true

row.status = status

this.temp = Object.assign({}, row) // copy obj

WebArticleApi.update(this.temp)

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

handleModifyIsDel(row, active) {

this.listLoading = true

row.isDel = active

this.temp = Object.assign({}, row) // copy obj

WebArticleApi.update(this.temp)

// 至于为什么不用以下方法，是因为懒得重写接口了，反正不是高并发项目……

// WebArticleApi.delSome(this.temp.id)

// 这里提供一个改进方案：新增一个包含id和active的对象，更新接口delSome为POST上传这个对象给后端做判断

// 说句实在话，目前这个方法也就是数据传输量多了点，后端bean拷贝的时候识别到一样的不会执行拷贝，所以效率也不会提高多少

this.listLoading = false

this.$message({

message: '操作成功',

type: 'success'

})

},

sortChange(data) {

const { prop, order } = data

console.log(order)

console.log(prop)

if (prop === 'date') {

this.sortByID(order)

}

},

sortByID(order) {

if (order === 'ascending') {

this.listQuery.direction = 'ASC'

} else {

this.listQuery.direction = 'DESC'

}

console.log(this.listQuery.direction)

this.handleFilter()

},

}

}

</script>

<template>

<div class="sidebar-logo-container" :class="{'collapse':collapse}">

<transition name="sidebarLogoFade">

<router-link v-if="collapse" key="collapse" class="sidebar-logo-link" to="/">

<img v-if="logo" :src="logo" class="sidebar-logo">

<h1 v-else class="sidebar-title">{{ title }} </h1>

</router-link>

<router-link v-else key="expand" class="sidebar-logo-link" to="/">

<img v-if="logo" :src="logo" class="sidebar-logo-ex">

<h1 class="sidebar-title">{{ title }} </h1>

</router-link>

</transition>

</div>

</template>

<script>

import logo from '@/assets/logo.png'

export default {

name: 'SidebarLogo',

props: {

collapse: {

type: Boolean,

required: true

}

},

data() {

return {

title: '米果传媒',

logo: logo

}

}

}

</script>

<style lang="scss" scoped>

.sidebarLogoFade-enter-active {

transition: opacity 1.5s;

}

.sidebarLogoFade-enter,

.sidebarLogoFade-leave-to {

opacity: 0;

}

.sidebar-logo-container {

position: relative;

width: 100%;

height: 100px;

line-height: 100px;

background: #2b2f3a;

text-align: center;

overflow: hidden;

& .sidebar-logo-link {

height: 100%;

width: 100%;

& .sidebar-logo-ex {

width: 60px;

height: 60px;

vertical-align: middle;

margin-right: 24px;

border-radius: 10px;

box-shadow: 0 0 15px white;

}

& .sidebar-logo {

width: 40px;

height: 40px;

vertical-align: middle;

margin-right: 24px;

border-radius: 10px;

box-shadow: 0 0 15px white;

}

& .sidebar-title {

display: inline-block;

margin: 0;

color: #fff;

font-weight: 600;

line-height: 50px;

font-size: 18px;

font-family: Avenir, Helvetica Neue, Arial, Helvetica, sans-serif;

vertical-align: middle;

}

}

&.collapse {

.sidebar-logo {

margin-right: 0px;

}

}

}

</style>

<template>

<div class="navbar">

<hamburger :is-active="sidebar.opened" class="hamburger-container" @toggleClick="toggleSideBar" />

<breadcrumb class="breadcrumb-container" />

<div class="right-menu">

<el-dropdown class="avatar-container" trigger="click">

<div class="avatar-wrapper">

<img src="../../assets/staff\_icon.png" class="user-avatar">

<i class="el-icon-caret-bottom" />

</div>

<el-dropdown-menu slot="dropdown" class="user-dropdown">

<router-link to="/">

<el-dropdown-item>

回到主页

</el-dropdown-item>

</router-link>

<a target="\_blank" href="http://migotimes.com/#/home">

<el-dropdown-item>前往官网</el-dropdown-item>

</a>

<el-dropdown-item>

<span style="display:block;" @click="handleEdit">修改信息</span>

</el-dropdown-item>

<el-dropdown-item divided>

<span style="display:block;" @click="logout">注销</span>

</el-dropdown-item>

</el-dropdown-menu>

</el-dropdown>

</div>

<el-dialog title="修改个人信息" :visible.sync="dialogFormVisible">

<el-form ref="dataForm" :rules="rules" :model="temp" label-position="left" label-width="100px" style="width: 400px; margin-left:50px;">

<el-form-item label="姓名" prop="name">

<el-input v-model="temp.name" />

</el-form-item>

<el-form-item label="昵称" prop="nickname">

<el-input v-model="temp.nickname" />

</el-form-item>

<el-form-item label="密码" prop="password">

<el-input v-model="temp.password" />

</el-form-item>

</el-form>

<div slot="footer" class="dialog-footer">

<el-button @click="dialogFormVisible = false">

取消

</el-button>

<el-button type="primary" @click="updateData">

确认

</el-button>

</div>

</el-dialog>

</div>

</template>

<script>

import { mapGetters } from 'vuex'

import Breadcrumb from '@/components/Breadcrumb'

import Hamburger from '@/components/Hamburger'

import LoginApi from '@/api/website/LoginApi'

export default {

components: {

Breadcrumb,

Hamburger

},

computed: {

...mapGetters([

'sidebar',

])

},

data() {

return {

temp: {

id: "",

name: "",

nickname: "",

password: "",

level: undefined

},

dialogFormVisible: false,

rules: {

name: ,

nickname: ,

password: ,

},

}

},

methods: {

handleEdit(){

LoginApi.getSelfInfo().then(data => {

this.temp.id = data.id

this.temp.name = data.name

this.temp.nickname = data.nickname

this.temp.password = data.password

this.temp.level = data.level

})

this.dialogFormVisible = true

},

updateData(){

LoginApi.setSelfInfo(this.temp).then(data => {

console.log(data)

})

this.dialogFormVisible = false

this.$notify({

title: 'success',

message: '创建成功',

type: 'success',

duration: 2000

})

},

toggleSideBar() {

this.$store.dispatch('app/toggleSideBar')

},

async logout() {

await LoginApi.logout().then(data => {

sessionStorage.setItem('user',null)

this.$store.dispatch('user/userLogout')

this.$store.dispatch('permission/clearRoutes')

}).catch(err => {

// debugger

console.log(err)

})

this.$router.push(`/login?redirect=${this.$route.fullPath}`)

}

}

}

</script>

<style lang="scss" scoped>

.navbar {

height: 50px;

overflow: hidden;

position: relative;

background: #fff;

box-shadow: 0 1px 4px rgba(0,21,41,.08);

.hamburger-container {

line-height: 46px;

height: 100%;

float: left;

cursor: pointer;

transition: background .3s;

-webkit-tap-highlight-color:transparent;

&:hover {

background: rgba(0, 0, 0, .025)

}

}

.breadcrumb-container {

float: left;

}

.right-menu {

float: right;

height: 100%;

line-height: 50px;

&:focus {

outline: none;

}

.right-menu-item {

display: inline-block;

padding: 0 8px;

height: 100%;

font-size: 18px;

color: #5a5e66;

vertical-align: text-bottom;

&.hover-effect {

cursor: pointer;

transition: background .3s;

&:hover {

background: rgba(0, 0, 0, .025)

}

}

}

.avatar-container {

margin-right: 30px;

.avatar-wrapper {

margin-top: 5px;

position: relative;

.user-avatar {

cursor: pointer;

width: 40px;

height: 40px;

border-radius: 10px;

}

.el-icon-caret-bottom {

cursor: pointer;

position: absolute;

right: -20px;

top: 25px;

font-size: 12px;

}

}

}

}

}

</style>

<template>

<el-dropdown :hide-on-click="false" :show-timeout="100" trigger="click">

<el-button plain>

Platfroms({{ platforms.length }})

<i class="el-icon-caret-bottom el-icon--right" />

</el-button>

<el-dropdown-menu slot="dropdown" class="no-border">

<el-checkbox-group v-model="platforms" style="padding: 5px 15px;">

<el-checkbox v-for="item in platformsOptions" :key="item.key" :label="item.key">

{{ item.name }}

</el-checkbox>

</el-checkbox-group>

</el-dropdown-menu>

</el-dropdown>

</template>

<script>

export default {

props: {

value: {

required: true,

default: () => ,

type: Array

}

},

data() {

return {

platformsOptions: [

{ key: 'a-platform', name: 'a-platform' },

{ key: 'b-platform', name: 'b-platform' },

{ key: 'c-platform', name: 'c-platform' }

]

}

},

computed: {

platforms: {

get() {

return this.value

},

set(val) {

this.$emit('input', val)

}

}

}

}

</script>

<template>

<div v-if="!item.hidden" class="menu-wrapper">

<template v-if="hasOneShowingChild(item.children,item) && (!onlyOneChild.children||onlyOneChild.noShowingChildren)&&!item.alwaysShow">

<app-link v-if="onlyOneChild.meta" :to="resolvePath(onlyOneChild.path)">

<el-menu-item :index="resolvePath(onlyOneChild.path)" :class="{'submenu-title-noDropdown':!isNest}">

<item :icon="onlyOneChild.meta.icon||(item.meta&&item.meta.icon)" :title="onlyOneChild.meta.title" />

</el-menu-item>

</app-link>

</template>

<el-submenu v-else ref="subMenu" :index="resolvePath(item.path)" popper-append-to-body>

<template slot="title">

<item v-if="item.meta" :icon="item.meta && item.meta.icon" :title="item.meta.title" />

</template>

<sidebar-item

v-for="child in item.children"

:key="child.path"

:is-nest="true"

:item="child"

:base-path="resolvePath(child.path)"

class="nest-menu"

/>

</el-submenu>

</div>

</template>

<script>

import path from 'path'

import { isExternal } from '@/utils/validate'

import Item from './Item'

import AppLink from './Link'

import FixiOSBug from './FixiOSBug'

export default {

name: 'SidebarItem',

components: { Item, AppLink },

mixins: ,

props: {

// route object

item: {

type: Object,

required: true

},

isNest: {

type: Boolean,

default: false

},

basePath: {

type: String,

default: ''

}

},

data() {

// To fix https://github.com/PanJiaChen/vue-admin-template/issues/237

// TODO: refactor with render function

this.onlyOneChild = null

return {}

},

methods: {

hasOneShowingChild(children = , parent) {

const showingChildren = children.filter(item => {

if (item.hidden) {

return false

} else {

// Temp set(will be used if only has one showing child)

this.onlyOneChild = item

return true

}

})

// When there is only one child router, the child router is displayed by default

if (showingChildren.length === 1) {

return true

}

// Show parent if there are no child router to display

if (showingChildren.length === 0) {

this.onlyOneChild = { ... parent, path: '', noShowingChildren: true }

return true

}

return false

},

resolvePath(routePath) {

if (isExternal(routePath)) {

return routePath

}

if (isExternal(this.basePath)) {

return this.basePath

}

return path.resolve(this.basePath, routePath)

}

}

}

</script>

<template>

<el-dropdown :show-timeout="100" trigger="click">

<el-button plain>

Link

<i class="el-icon-caret-bottom el-icon--right" />

</el-button>

<el-dropdown-menu slot="dropdown" class="no-padding no-border" style="width:400px">

<el-form-item label-width="0px" style="margin-bottom: 0px" prop="source\_uri">

<el-input v-model="source\_uri" placeholder="Please enter the content">

<template slot="prepend">

URL

</template>

</el-input>

</el-form-item>

</el-dropdown-menu>

</el-dropdown>

</template>

<script>

export default {

props: {

value: {

type: String,

default: ''

}

},

computed: {

source\_uri: {

get() {

return this.value

},

set(val) {

this.$emit('input', val)

}

}

}

}

</script>

<template>

<aside>

Creating and editing pages cannot be cached by keep-alive because keep-alive include does not currently support

caching based on routes, so it is currently cached based on component name. If you want to achieve a similar caching

effect, you can use a browser caching scheme such as localStorage. Or do not use keep-alive include to cache all

pages directly. See details

<a

href="https://panjiachen.github.io/vue-element-admin-site/guide/essentials/tags-view.html"

target="\_blank"

>Document</a>

</aside>

</template>

// https://eslint.org/docs/user-guide/configuring

module.exports = {

root: true,

parserOptions: {

parser: 'babel-eslint'

},

env: {

browser: true,

},

extends: [

// https://github.com/vuejs/eslint-plugin-vue#priority-a-essential-error-prevention

// consider switching to `plugin:vue/strongly-recommended` or `plugin:vue/recommended` for stricter rules.

'plugin:vue/essential',

// https://github.com/standard/standard/blob/master/docs/RULES-en.md

'standard'

],

// required to lint \*.vue files

plugins: [

'vue'

],

// add your custom rules here

rules: {

// allow async-await

'generator-star-spacing': 'off',

// allow debugger during development

'no-debugger': process.env.NODE\_ENV === 'production' ? 'error' : 'off'

}

}

// https://github.com/michael-ciniawsky/postcss-load-config

module.exports = {

"plugins": {

"postcss-import": {},

"postcss-url": {},

// to edit target browsers: use "browserslist" field in package.json

"autoprefixer": {}

}

}

import Cookies from 'js-cookie'

const state = {

sidebar: {

opened: Cookies.get('sidebarStatus') ? !!+Cookies.get('sidebarStatus') : true,

withoutAnimation: false

},

device: 'desktop',

}

const mutations = {

TOGGLE\_SIDEBAR: state => {

state.sidebar.opened = !state.sidebar.opened

state.sidebar.withoutAnimation = false

if (state.sidebar.opened) {

Cookies.set('sidebarStatus', 1)

} else {

Cookies.set('sidebarStatus', 0)

}

},

CLOSE\_SIDEBAR: (state, withoutAnimation) => {

Cookies.set('sidebarStatus', 0)

state.sidebar.opened = false

state.sidebar.withoutAnimation = withoutAnimation

},

TOGGLE\_DEVICE: (state, device) => {

state.device = device

}

}

const actions = {

toggleSideBar({ commit }) {

commit('TOGGLE\_SIDEBAR')

},

closeSideBar({ commit }, { withoutAnimation }) {

commit('CLOSE\_SIDEBAR', withoutAnimation)

},

toggleDevice({ commit }, device) {

commit('TOGGLE\_DEVICE', device)

}

}

export default {

namespaced: true,

state,

mutations,

actions

}

import Cookies from 'js-cookie'

const TokenKey = 'vue\_admin\_template\_token'

export function getToken() {

return Cookies.get(TokenKey)

}

export function setToken(token) {

return Cookies.set(TokenKey, token)

}

export function removeToken() {

return Cookies.remove(TokenKey)

}

'use strict'

require('./check-versions')()

process.env.NODE\_ENV = 'production'

const ora = require('ora')

const rm = require('rimraf')

const path = require('path')

const chalk = require('chalk')

const webpack = require('webpack')

const config = require('../config')

const webpackConfig = require('./webpack.prod.conf')

const spinner = ora('building for production...')

spinner.start()

rm(path.join(config.build.assetsRoot, config.build.assetsSubDirectory), err => {

if (err) throw err

webpack(webpackConfig, (err, stats) => {

spinner.stop()

if (err) throw err

process.stdout.write(stats.toString({

colors: true,

modules: false,

children: false, // If you are using ts-loader, setting this to true will make TypeScript errors show up during build.

chunks: false,

chunkModules: false

}) + '\n\n')

if (stats.hasErrors()) {

console.log(chalk.red(' Build failed with errors.\n'))

process.exit(1)

}

console.log(chalk.cyan(' Build complete.\n'))

console.log(chalk.yellow(

' Tip: built files are meant to be served over an HTTP server.\n' +

' Opening index.html over file:// won\'t work.\n'

))

})

})

'use strict'

const chalk = require('chalk')

const semver = require('semver')

const packageConfig = require('../package.json')

const shell = require('shelljs')

function exec (cmd) {

return require('child\_process').execSync(cmd).toString().trim()

}

const versionRequirements = [

{

name: 'node',

currentVersion: semver.clean(process.version),

versionRequirement: packageConfig.engines.node

}

]

if (shell.which('npm')) {

versionRequirements.push({

name: 'npm',

currentVersion: exec('npm --version'),

versionRequirement: packageConfig.engines.npm

})

}

module.exports = function () {

const warnings =

for (let i = 0; i < versionRequirements.length; i++) {

const mod = versionRequirements

if (!semver.satisfies(mod.currentVersion, mod.versionRequirement)) {

warnings.push(mod.name + ': ' +

chalk.red(mod.currentVersion) + ' should be ' +

chalk.green(mod.versionRequirement)

)

}

}

if (warnings.length) {

console.log('')

console.log(chalk.yellow('To use this template, you must update following to modules:'))

console.log()

for (let i = 0; i < warnings.length; i++) {

const warning = warnings

console.log(' ' + warning)

}

console.log()

process.exit(1)

}

}

/\*eslint-disable\*/

// 把 YYYY-MM-DD 改成了 yyyy-MM-dd

(function (main) {

'use strict';

/\*\*

\* Parse or format dates

\* @class fecha

\*/

var fecha = {};

var token = /d{1,4}|M{1,4}|yy(?:yy)?|S{1,3}|Do|ZZ|()\1?||"\*"|'\*'/g;

var twoDigits = /\d\d?/;

var threeDigits = /\d{3}/;

var fourDigits = /\d{4}/;

var word = /\*+|+(\s\*?+){1,2}/i;

var noop = function () {

};

function shorten(arr, sLen) {

var newArr = ;

for (var i = 0, len = arr.length; i < len; i++) {

newArr.push(arr.substr(0, sLen));

}

return newArr;

}

function monthUpdate(arrName) {

return function (d, v, i18n) {

var index = i18n.indexOf(v.charAt(0).toUpperCase() + v.substr(1).toLowerCase());

if (~index) {

d.month = index;

}

};

}

function pad(val, len) {

val = String(val);

len = len || 2;

while (val.length < len) {

val = '0' + val;

}

return val;

}

var dayNames = ;

var monthNames = ;

var monthNamesShort = shorten(monthNames, 3);

var dayNamesShort = shorten(dayNames, 3);

fecha.i18n = {

dayNamesShort: dayNamesShort,

dayNames: dayNames,

monthNamesShort: monthNamesShort,

monthNames: monthNames,

amPm: ,

DoFn: function DoFn(D) {

return D + ;

}

};

var formatFlags = {

D: function (dateObj) {

return dateObj.getDay();

},

DD: function (dateObj) {

return pad(dateObj.getDay());

},

Do: function (dateObj, i18n) {

return i18n.DoFn(dateObj.getDate());

},

d: function (dateObj) {

return dateObj.getDate();

},

dd: function (dateObj) {

return pad(dateObj.getDate());

},

ddd: function (dateObj, i18n) {

return i18n.dayNamesShort;

},

dddd: function (dateObj, i18n) {

return i18n.dayNames;

},

M: function (dateObj) {

return dateObj.getMonth() + 1;

},

MM: function (dateObj) {

return pad(dateObj.getMonth() + 1);

},

MMM: function (dateObj, i18n) {

return i18n.monthNamesShort;

},

MMMM: function (dateObj, i18n) {

return i18n.monthNames;

},

yy: function (dateObj) {

return String(dateObj.getFullYear()).substr(2);

},

yyyy: function (dateObj) {

return dateObj.getFullYear();

},

h: function (dateObj) {

return dateObj.getHours() % 12 || 12;

},

hh: function (dateObj) {

return pad(dateObj.getHours() % 12 || 12);

},

H: function (dateObj) {

return dateObj.getHours();

},

HH: function (dateObj) {

return pad(dateObj.getHours());

},

m: function (dateObj) {

return dateObj.getMinutes();

},

mm: function (dateObj) {

return pad(dateObj.getMinutes());

},

s: function (dateObj) {

return dateObj.getSeconds();

},

ss: function (dateObj) {

return pad(dateObj.getSeconds());

},

S: function (dateObj) {

return Math.round(dateObj.getMilliseconds() / 100);

},

SS: function (dateObj) {

return pad(Math.round(dateObj.getMilliseconds() / 10), 2);

},

SSS: function (dateObj) {

return pad(dateObj.getMilliseconds(), 3);

},

a: function (dateObj, i18n) {

return dateObj.getHours() < 12 ? i18n.amPm : i18n.amPm;

},

A: function (dateObj, i18n) {

return dateObj.getHours() < 12 ? i18n.amPm.toUpperCase() : i18n.amPm.toUpperCase();

},

ZZ: function (dateObj) {

var o = dateObj.getTimezoneOffset();

return (o > 0 ? '-' : '+') + pad(Math.floor(Math.abs(o) / 60) \* 100 + Math.abs(o) % 60, 4);

}

};

var parseFlags = {

d: [twoDigits, function (d, v) {

d.day = v;

}],

M: [twoDigits, function (d, v) {

d.month = v - 1;

}],

yy: [twoDigits, function (d, v) {

var da = new Date(), cent = +('' + da.getFullYear()).substr(0, 2);

d.year = '' + (v > 68 ? cent - 1 : cent) + v;

}],

h: [twoDigits, function (d, v) {

d.hour = v;

}],

m: [twoDigits, function (d, v) {

d.minute = v;

}],

s: [twoDigits, function (d, v) {

d.second = v;

}],

yyyy: [fourDigits, function (d, v) {

d.year = v;

}],

S: [/\d/, function (d, v) {

d.millisecond = v \* 100;

}],

SS: [/\d{2}/, function (d, v) {

d.millisecond = v \* 10;

}],

SSS: [threeDigits, function (d, v) {

d.millisecond = v;

}],

D: ,

ddd: ,

MMM: ,

MMMM: ,

a: [word, function (d, v, i18n) {

var val = v.toLowerCase();

if (val === i18n.amPm) {

d.isPm = false;

} else if (val === i18n.amPm) {

d.isPm = true;

}

}],

ZZ: \d\d:?\d\d/, function (d, v) {

var parts = (v + '').match(/(|\d\d)/gi), minutes;

if (parts) {

minutes = +(parts \* 60) + parseInt(parts, 10);

d.timezoneOffset = parts === '+' ? minutes : -minutes;

}

}]

};

parseFlags.DD = parseFlags.DD;

parseFlags.dddd = parseFlags.ddd;

parseFlags.Do = parseFlags.dd = parseFlags.d;

parseFlags.mm = parseFlags.m;

parseFlags.hh = parseFlags.H = parseFlags.HH = parseFlags.h;

parseFlags.MM = parseFlags.M;

parseFlags.ss = parseFlags.s;

parseFlags.A = parseFlags.a;

// Some common format strings

fecha.masks = {

'default': 'ddd MMM dd yyyy HH:mm:ss',

shortDate: 'M/D/yy',

mediumDate: 'MMM d, yyyy',

longDate: 'MMMM d, yyyy',

fullDate: 'dddd, MMMM d, yyyy',

shortTime: 'HH:mm',

mediumTime: 'HH:mm:ss',

longTime: 'HH:mm:ss.SSS'

};

/\*\*\*

\* Format a date

\* @method format

\* @param {Date|number} dateObj

\* @param {string} mask Format of the date, i.e. 'mm-dd-yy' or 'shortDate'

\*/

fecha.format = function (dateObj, mask, i18nSettings) {

var i18n = i18nSettings || fecha.i18n;

if (typeof dateObj === 'number') {

dateObj = new Date(dateObj);

}

if (Object.prototype.toString.call(dateObj) !== '' || isNaN(dateObj.getTime())) {

throw new Error('Invalid Date in fecha.format');

}

mask = fecha.masks || mask || fecha.masks;

return mask.replace(token, function ($0) {

return $0 in formatFlags ? formatFlags(dateObj, i18n) : $0.slice(1, $0.length - 1);

});

};

/\*\*

\* Parse a date string into an object, changes - into /

\* @method parse

\* @param {string} dateStr Date string

\* @param {string} format Date parse format

\* @returns {Date|boolean}

\*/

fecha.parse = function (dateStr, format, i18nSettings) {

var i18n = i18nSettings || fecha.i18n;

if (typeof format !== 'string') {

throw new Error('Invalid format in fecha.parse');

}

format = fecha.masks || format;

// Avoid regular expression denial of service, fail early for really long strings

// https://www.owasp.org/index.php/Regular\_expression\_Denial\_of\_Service\_-\_ReDoS

if (dateStr.length > 1000) {

return false;

}

var isValid = true;

var dateInfo = {};

format.replace(token, function ($0) {

if (parseFlags) {

var info = parseFlags;

var index = dateStr.search(info);

if (!~index) {

isValid = false;

} else {

dateStr.replace(info, function (result) {

info(dateInfo, result, i18n);

dateStr = dateStr.substr(index + result.length);

return result;

});

}

}

return parseFlags ? '' : $0.slice(1, $0.length - 1);

});

if (!isValid) {

return false;

}

var today = new Date();

if (dateInfo.isPm === true && dateInfo.hour != null && +dateInfo.hour !== 12) {

dateInfo.hour = +dateInfo.hour + 12;

} else if (dateInfo.isPm === false && +dateInfo.hour === 12) {

dateInfo.hour = 0;

}

var date;

if (dateInfo.timezoneOffset != null) {

dateInfo.minute = +(dateInfo.minute || 0) - +dateInfo.timezoneOffset;

date = new Date(Date.UTC(dateInfo.year || today.getFullYear(), dateInfo.month || 0, dateInfo.day || 1,

dateInfo.hour || 0, dateInfo.minute || 0, dateInfo.second || 0, dateInfo.millisecond || 0));

} else {

date = new Date(dateInfo.year || today.getFullYear(), dateInfo.month || 0, dateInfo.day || 1,

dateInfo.hour || 0, dateInfo.minute || 0, dateInfo.second || 0, dateInfo.millisecond || 0);

}

return date;

};

/\* istanbul ignore next \*/

if (typeof module !== 'undefined' && module.exports) {

module.exports = fecha;

} else if (typeof define === 'function' && define.amd) {

define(function () {

return fecha;

});

} else {

main.fecha = fecha;

}

})(this);

'use strict'

const merge = require('webpack-merge')

const prodEnv = require('./prod.env')

module.exports = merge(prodEnv, {

NODE\_ENV: '"development"',

HOME\_URL: '"http://localhost:9090"'

})

export default {

computed: {

device() {

return this.$store.state.app.device

}

},

mounted() {

// In order to fix the click on menu on the ios device will trigger the mouseleave bug

// https://github.com/PanJiaChen/vue-element-admin/issues/1135

this.fixBugIniOS()

},

methods: {

fixBugIniOS() {

const $subMenu = this.$refs.subMenu

if ($subMenu) {

const handleMouseleave = $subMenu.handleMouseleave

$subMenu.handleMouseleave = (e) => {

if (this.device === 'mobile') {

return

}

handleMouseleave(e)

}

}

}

}

}

import defaultSettings from '@/settings'

const title = defaultSettings.title || 'Vue Admin Template'

export default function getPageTitle(pageTitle) {

if (pageTitle) {

return `${pageTitle} - ${title}`

}

return `${title}`

}

const getters = {

sidebar: state => state.app.sidebar,

device: state => state.app.device,

roles: state => state.user.roles,

has\_routes: state => state.permission.has\_routes,

permission\_routes: state => state.permission.routes,

visitedViews: state => state.tagsView.visitedViews,

cachedViews: state => state.tagsView.cachedViews,

}

export default getters

import {Message} from 'element-ui'

import axios from 'axios'

import store from '../store'

import router from '../router'

axios.defaults.withCredentials=true

axios.defaults.baseURL = process.env.HOME\_URL

axios.defaults.timeout = 30 \* 1000 // 设置接口响应时间

/\*\*

\* 功能描述：Http Request 拦截器

\*/

axios.interceptors.request.use((config) => {

return config

}, (error) => { // 请求错误时做些事(接口错误、超时等)

Message.error(`请求参数错误`, 10)

return Promise.reject(error) // 在调用的那边可以拿到(catch)你想返回的错误信息

})

/\*\*

\* 功能描述：Http Response 拦截器

\*/

axios.interceptors.response.use((res) => {

let isTips = res.config.showMessage === undefined || res.config.showMessage === true

let data = res.data

// console.log(res)

/\* 根据返回的code值来做不同的处理（和后端约定） \*/

switch (data.code) {

case 200:

return Promise.resolve(data)

case 401:

// console.log("32454354")

return Promise.resolve(data)

default:

isTips && Message.error(`服务器返回异常：${data.message}`, 10)

return Promise.reject(res)

}

}, (error) => {

// Message.error(`服务器返回异常：${error}`, 10)

if (error && error.response) {

switch (error.response.status) {

case 400:

break

case 401:

// console.log("false")

Message.warning('还没登录或账号已经过期')

sessionStorage.setItem('user',null)

store.dispatch('user/userLogout').then(r => {})

store.dispatch('permission/clearRoutes').then(r => {})

router.push(`/login`)

break

default:

break

}

}

return Promise.reject(error)

})

export default {

get (url, data = {}) {

return new Promise((resolve, reject) => {

axios.get(url, data).then(response => {

resolve(response)

}).catch(err => {

reject(err)

})

})

},

delete (url, data = {}) {

return new Promise((resolve, reject) => {

axios.delete(url, data).then(response => {

resolve(response)

}).catch(err => {

reject(err)

})

})

},

post (url, data = {}) {

return new Promise((resolve, reject) => {

axios.post(url, data).then(response => {

resolve(response)

}, err => {

reject(err)

})

})

},

put (url, data = {}) {

return new Promise((resolve, reject) => {

axios.put(url, data).then(response => {

resolve(response)

}, err => {

reject(err)

})

})

},

patch (url, data = {}) {

return new Promise((resolve, reject) => {

axios.patch(url, data).then(response => {

resolve(response)

}, err => {

reject(err)

})

})

}

}

export { default as CommentDropdown } from './Comment'

export { default as PlatformDropdown } from './Platform'

export { default as SourceUrlDropdown } from './SourceUrl'

import waves from './waves'

const install = function(Vue) {

Vue.directive('waves', waves)

}

if (window.Vue) {

window.waves = waves

Vue.use(install); // eslint-disable-line

}

waves.install = install

export default waves

import Vue from 'vue'

import SvgIcon from '@/components/SvgIcon'// svg component

// register globally

Vue.component('svg-icon', SvgIcon)

const req = require.context('./svg', false, /\.svg$/)

const requireAll = requireContext => requireContext.keys().map(requireContext)

requireAll(req)

export { default as Navbar } from './Navbar'

export { default as Sidebar } from './Sidebar'

export { default as AppMain } from './AppMain'

import Vue from 'vue'

import Router from 'vue-router'

Vue.use(Router)

/\* Layout \*/

import Layout from '@/layout'

/\*\*

\* 注意：子菜单仅在路由children.length> = 1时出现

\* 详细信息请参见：https://panjiachen.github.io/vue-element-admin-site/guide/essentials/router-and-nav.html

\*

\* 隐藏：如果设置为true，则为true，则项目不会显示在边栏中（默认为false）

\* alwaysShow：如果设置为true，则将始终显示根菜单

\* 如果未设置alwaysShow，则当项有多个子路线时，

\* 它将变为嵌套模式，否则不显示根菜单

\* redirect：noRedirect如果设置了noRedirect将不会在面包屑中重定向

\* name：“路由器名称”，该名称由<keep-alive>使用（必须设置！！！）

\* meta：{

角色：控制页面角色（您可以设置多个角色）

内容分发后端代码

# -\*- coding: utf-8 -\*-

"""

:author: comwrg

:license: MIT

:time: 2017/06/09

"""

import base64

import hashlib

import logging

import math

import os

import re

import requests

import rsa

import time

from io import BufferedReader

from typing import \*

from urllib import parse

from requests.adapters import HTTPAdapter

from urllib3 import Retry

log = logging.getLogger(\_\_name\_\_)

log.addHandler(logging.NullHandler())

class VideoPart:

def \_\_init\_\_(self, path, title='', desc=''):

self.path = path

self.title = title

self.desc = desc

def \_\_repr\_\_(self):

return '<{clazz}, path: {path}, title: {title}, desc: {desc}>'.format(clazz=self.\_\_class\_\_.\_\_name\_\_, path=self.path, title=self.title, desc=self.desc)

class Bilibili:

def \_\_init\_\_(self, cookie=None):

self.session = requests.session()

# debug

def debug\_response(r, \*args, \*\*kwargs):

log.debug(r.text)

self.session.hooks = {'response': debug\_response}

#

if cookie:

self.session.headers["cookie"] = cookie

self.csrf = re.search('bili\_jct=(.\*?)(;|$)', cookie).group(1)

self.mid = re.search('DedeUserID=(.\*?)(;|$)', cookie).group(1)

self.session.headers['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01'

self.session.headers['Referer'] = 'https://space.bilibili.com/{mid}/#!/'.format(mid=self.mid)

# session.headers['User-Agent'] = 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/57.0.2987.133 Safari/537.36'

# session.headers['Content-Type'] = 'application/x-www-form-urlencoded; charset=UTF-8'

def login(self, user, pwd):

"""

:param user: username

:type user: str

:param pwd: password

:type pwd: str

:return: if success return True

else raise Exception

"""

APPKEY = '4409e2ce8ffd12b8'

ACTIONKEY = 'appkey'

BUILD = 101800

DEVICE = 'android\_tv\_yst'

MOBI\_APP = 'android\_tv\_yst'

PLATFORM = 'android'

APPSECRET = '59b43e04ad6965f34319062b478f83dd'

def md5(s):

h = hashlib.md5()

h.update(s.encode('utf-8'))

return h.hexdigest()

def sign(s):

"""

:return: return sign

"""

return md5(s + APPSECRET)

def signed\_body(body):

"""

:return: body which be added sign

"""

if isinstance(body, str):

return body + '&sign=' + sign(body)

elif isinstance(body, dict):

ls = []

for k, v in body.items():

ls.append(k + '=' + v)

body['sign'] = sign('&'.join(ls))

return body

def getkey():

"""

:return: hash, key

"""

r = self.session.post(

'https://passport.bilibili.com/api/oauth2/getKey',

signed\_body({'appkey': APPKEY}),

)

# {"ts":1544152439,"code":0,"data":{"hash":"99c7573759582e0b","key":"-----BEGIN PUBLIC----- -----END PUBLIC KEY-----\n"}}

json = r.json()

data = json['data']

return data['hash'], data['key']

def access\_token\_2\_cookie(access\_token):

r = self.session.get(

'https://passport.bilibili.com/api/login/sso?' + \

signed\_body(

'access\_key={access\_token}&appkey={appkey}&gourl=https%3A%2F%2Faccount.bilibili.com%2Faccount%2Fhome'

.format(access\_token=access\_token, appkey=APPKEY),

),

allow\_redirects=False,

)

return r.cookies.get\_dict(domain=".bilibili.com")

self.session.headers['Content-Type'] = 'application/x-www-form-urlencoded; charset=UTF-8'

h, k = getkey()

pwd = base64.b64encode(

rsa.encrypt(

(h + pwd).encode('utf-8'),

rsa.PublicKey.load\_pkcs1\_openssl\_pem(k.encode()),

),

)

user = parse.quote\_plus(user)

pwd = parse.quote\_plus(pwd)

r = self.session.post(

'https://passport.snm0516.aisee.tv/api/tv/login',

signed\_body(

'appkey={appkey}&build={build}&captcha=&channel=master&'

'guid=XYEBAA3E54D502E37BD606F0589A356902FCF&mobi\_app={mobi\_app}&'

'password={password}&platform={platform}&token=5598158bcd8511e2&ts=0&username={username}'

.format(appkey=APPKEY, build=BUILD, platform=PLATFORM, mobi\_app=MOBI\_APP, username=user, password=pwd)),

)

json = r.json()

if json['code'] == -105:

# need captcha

raise Exception('TODO: login with captcha')

if json['code'] != 0:

raise Exception(r.text)

access\_token = json['data']['token\_info']['access\_token']

cookie\_dict = access\_token\_2\_cookie(access\_token)

cookie = '; '.join(

'%s=%s' % (k, v)

for k, v in cookie\_dict.items()

)

self.session.headers["cookie"] = cookie

self.csrf = re.search('bili\_jct=(.\*?)(;|$)', cookie).group(1)

self.mid = re.search('DedeUserID=(.\*?)(;|$)', cookie).group(1)

self.session.headers['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01'

self.session.headers['Referer'] = 'https://space.bilibili.com/{mid}/#!/'.format(mid=self.mid)

return True

def upload(self,

parts: Union[VideoPart, List[VideoPart]],

title: str,

tid: int,

tag: List[str],

desc: str,

source='',

cover='',

no\_reprint: bool = True,

dynamic='',

dtime=None,

open\_elec: bool = True,

open\_subtitle: bool = True,

max\_retry=5,

):

"""

:param parts: e.g. VideoPart('part path', 'part title', 'part desc'), or [VideoPart(...), VideoPart(...)]

:type parts: Union[VideoPart, List[VideoPart]]

:param title: video's title

:type title: str

:param tid: video type, see: https://member.bilibili.com/x/web/archive/pre

or https://github.com/uupers/BiliSpider/wiki/%E8%A7%86%E9%A2%91%E5%88%86%E5%8C%BA%E5%AF%B9%E5%BA%94%E8%A1%A8

:type tid: int

:param tag: video's tag

:type tag: List[str]

:param desc: video's description

:type desc: str

:param dtime: (optional) publish date timestamp (10 digits Unix timestamp e.g. 1551533438)

:type dtime: int

:param source: (optional) 转载地址

:type source: str

:param cover: (optional) cover's URL, use method \*cover\_up\* to get

:type cover: str

:param no\_reprint: (optional) Is reprint allowed

:type no\_reprint: bool

:param dynamic: 粉丝动态

:type dynamic: str

:param open\_elec: (optional) whether to open charging panel (充电面板)

:type open\_elec: bool

:param open\_subtitle: (optional) Is uploading subtitles allowed

:type open\_subtitle: bool

:param max\_retry: (optional) max retry times per chunk

:type max\_retry: int

"""

if len(title) > 80:

raise Exception("标题长度超过80字")

if len(source) > 200:

raise Exception("转载地址长度超过200字")

self.session.headers['Content-Type'] = 'application/json; charset=utf-8'

if not isinstance(parts, list):

parts = [parts]

# retry by status

retries = Retry(

total=max\_retry,

backoff\_factor=1,

status\_forcelist=(504, ),

)

self.session.mount('https://', HTTPAdapter(max\_retries=retries))

self.session.mount('http://', HTTPAdapter(max\_retries=retries))

#

videos = []

for part in parts:

filepath = part.path

filename = os.path.basename(filepath)

filesize = os.path.getsize(filepath)

r = self.session.get('https://member.bilibili.com/preupload?'

'os=upos&upcdn=ws&name={name}&size={size}&r=upos&profile=ugcupos%2Fyb&ssl=0'

.format(name=parse.quote\_plus(filename), size=filesize))

"""return example

{

"upos\_uri": "upos://ugc/i181012ws18x52mti3gg0h33chn3tyhp.mp4",

"biz\_id": 58993125,

"endpoint": "//upos-hz-upcdnws.acgvideo.com",

"endpoints": [

"//upos-hz-upcdnws.acgvideo.com",

"//upos-hz-upcdntx.acgvideo.com"

],

"chunk\_retry\_delay": 3,

"chunk\_retry": 200,

"chunk\_size": 4194304,

"threads": 2,

"timeout": 900,

"auth": "os=upos&cdn=upcdnws&uid=&net\_state=4&device=&build=&os\_version=&ak=×tamp=&sign=",

"OK": 1

}

"""

json = r.json()

upos\_uri = json['upos\_uri']

endpoint = json['endpoint']

auth = json['auth']

biz\_id = json['biz\_id']

chunk\_size = json['chunk\_size']

self.session.headers['X-Upos-Auth'] = auth # add auth header

r = self.session.post('https:{}/{}?uploads&output=json'.format(endpoint, upos\_uri.replace('upos://', '')))

# {"upload\_id":"72eb747b9650b8c7995fdb0efbdc2bb6","key":"\/i181012ws2wg1tb7tjzswk2voxrwlk1u.mp4","OK":1,"bucket":"ugc"}

json = r.json()

upload\_id = json['upload\_id']

with open(filepath, 'rb') as f:

chunks\_num = math.ceil(filesize / chunk\_size)

chunks\_index = -1

while True:

chunks\_data = f.read(chunk\_size)

if not chunks\_data:

break

chunks\_index += 1 # start with 0

def upload\_chunk():

r = self.session.put('https:{endpoint}/{upos\_uri}?'

'partNumber={part\_number}&uploadId={upload\_id}&chunk={chunk}&chunks={chunks}&size={size}&start={start}&end={end}&total={total}'

.format(endpoint=endpoint,

upos\_uri=upos\_uri.replace('upos://', ''),

part\_number=chunks\_index+1, # starts with 1

upload\_id=upload\_id,

chunk=chunks\_index,

chunks=chunks\_num,

size=len(chunks\_data),

start=chunks\_index \* chunk\_size,

end=chunks\_index \* chunk\_size + len(chunks\_data),

total=filesize,

),

chunks\_data,

)

return r

def retry\_upload\_chunk():

"""return :class:`Response` if upload success, else return None."""

for i in range(max\_retry):

r = upload\_chunk()

if r.status\_code == 200:

return r

log.info(r.text)

log.info('{}/{} retry stage {}/{}'.format(chunks\_index, chunks\_num, i, max\_retry))

log.info('sleep %ds', 5 \* i)

time.sleep(5 \* i)

return None

r = retry\_upload\_chunk()

if r:

log.info('upload part {}/{}'.format(chunks\_index, chunks\_num))

else:

raise Exception('upload reach max retry times at part {}/{}'.format(chunks\_index, chunks\_num))

# NOT DELETE! Refer to https://github.com/comwrg/bilibiliupload/issues/15#issuecomment-424379769

self.session.post('https:{endpoint}/{upos\_uri}?'

'output=json&name={name}&profile=ugcupos%2Fyb&uploadId={upload\_id}&biz\_id={biz\_id}'

.format(endpoint=endpoint,

upos\_uri=upos\_uri.replace('upos://', ''),

name=filename,

upload\_id=upload\_id,

biz\_id=biz\_id,

),

{"parts": [{"partNumber": i, "eTag": "etag"} for i in range(1, chunks\_num+1)]},

)

videos.append({'filename': upos\_uri.replace('upos://ugc/', '').split('.')[0],

'title' : part.title,

'desc' : part.desc})

# if source is empty, copyright=1, else copyright=2

copyright = 2 if source else 1

def add():

r = self.session.post('https://member.bilibili.com/x/vu/web/add?csrf=' + self.csrf,

json={

"copyright" : copyright,

"source" : source,

"title" : title,

"tid" : tid,

"tag" : ','.join(tag),

"no\_reprint": int(no\_reprint),

"desc" : desc,

"cover" : cover,

"mission\_id": 0,

"order\_id" : 0,

"videos" : videos,

"dtime" : dtime,

"open\_elec" : int(open\_elec),

"dynamic" : dynamic,

"subtitle" : {

"lan" : "",

"open": int(open\_subtitle),

},

},

)

return r

def retry\_add():

for i in range(max\_retry):

r = add()

json = r.json()

code = json['code']

if code == 0:

return r

# {"code":20001,"message":"投稿服务异常","ttl":1}

if code in (20001, ):

log.info('retry add video {}/{}, {}'.format(i, max\_retry, r.text))

else:

raise Exception('Fail to add video, {}'.format(r.text))

log.info('sleep %ds', 5 \* i)

time.sleep(5 \* i)

raise Exception('Add video reach max retry times.')

r = retry\_add()

return r.json()

def addChannel(self, name, intro=''):

"""

:param name: channel's name

:type name: str

:param intro: channel's introduction

:type intro: str

"""

r = self.session.post(

url='https://api.bilibili.com/x/space/channel/add',

data={

'name' : name,

'intro': intro,

'jsonp' : 'jsonp',

'csrf' : self.csrf,

},

# name=123&intro=123&aids=&csrf=565d7ed17cef2cc8ad054210c4e64324&\_=1497077610768

)

# return

# {"status":true,"data":{"cid":"15812"}}

def channel\_addVideo(self, cid, aids):

"""

:param cid: channel's id

:type cid: int

:param aids: videos' id

:type aids: list<int>

"""

r = self.session.post(

url='https://api.bilibili.com/x/space/channel/video/add',

data={

'aids': '%2C'.join(aids),

'cid' : cid,

'csrf': self.csrf,

},

# aids=9953555%2C9872953&cid=15814&csrf=565d7ed17cef2cc8ad054210c4e64324&\_=1497079332679

)

def cover\_up(self, img: Union[str, BufferedReader]):

"""

:param img: img path or stream

:return: img URL

"""

if isinstance(img, str):

f = open(img, 'rb')

else:

f = img

self.session.headers['Content-Type'] = 'application/x-www-form-urlencoded'

r = self.session.post(

url='https://member.bilibili.com/x/vu/web/cover/up',

data={

'cover': b'data:image/jpeg;base64,' + (base64.b64encode(f.read())),

'csrf': self.csrf,

},

)

# {"code":0,"data":{"url":"http://i0.hdslb.com/bfs/archive/67db4a6eae398c309244e74f6e85ae8d813bd7c9.jpg"},"message":"","ttl":1}

return r.json()['data']['url']

import pika,os,json,re

from bilibiliupload import \*

import MySQLdb

class MIGO(object):

# 数据库连接模块

# def \_\_init\_\_(self):

# self.migo\_db = MySQLdb.connect("localhost","root","b66a33eca6314e6a","miguo\_matrix")

# self.cursor = self.migo\_db.cursor()

def run(self):

credentials = pika.PlainCredentials('Hocassian', 'Hechenghan1234')

connection = pika.BlockingConnection(pika.ConnectionParameters(host = 'localhost',port = 5672,virtual\_host = '/upload',credentials = credentials))

channel = connection.channel()

# 创建临时队列，队列名传空字符，consumer关闭后，队列自动删除

channel.queue\_declare(queue='videoSendingQueue',durable = True)

channel.exchange\_declare(exchange = 'videoSendingExchange',durable = True, exchange\_type='direct')

# 绑定exchange和队列 exchange 使我们能够确切地指定消息应该到哪个队列去

channel.queue\_bind(exchange = 'videoSendingExchange', queue='videoSendingQueue', routing\_key='videoSendingRouting')

# 设置成 False，在调用callback函数时，未收到确认标识，消息会重回队列。True，无论调用callback成功与否，消息都被消费掉

channel.basic\_consume('videoSendingQueue',self.callback, auto\_ack = False)

channel.start\_consuming()

# 定义一个回调函数来处理消息队列中的消息，这里是打印出来

def callback(self, ch, method, properties, body):

ch.basic\_ack(delivery\_tag = method.delivery\_tag)

print(body.decode())

# video\_info\_json = body.decode()

video\_info\_dict = json.loads(body.decode())

\_video\_callback = self.video\_upload(video\_info\_dict)

video\_callback = str(\_video\_callback)

# 测试值 \_video\_callback = "{'code': 0, 'message': '0', 'ttl': 1, 'data': {'aid': 980000000, 'bvid': 'BV1LE411P7vz'}}"

video\_callback\_dict = json.loads(video\_callback.replace('\'','\"'))

print(\_video\_callback)

self.sql\_callback(video\_callback\_dict, video\_info\_dict)

def video\_upload(self, video\_info):

b = Bilibili()

login\_status = b.login('18026137881','zxd;+mgH')

print("登陆情况", login\_status)

# tags = list(video\_info['videoTag'])

print(video\_info['messageData']['videoTag'])

tags = video\_info['messageData']['videoTag']

abs\_path = './file/'

video\_upload\_path = os.path.join(abs\_path, video\_info['messageData']['videoPath'])

pic\_upload\_path = os.path.join(abs\_path, video\_info['messageData']['videoPic'])

videoPart = VideoPart(video\_upload\_path, video\_info['messageData']['videoTitle'])

print(video\_upload\_path)

video\_callback = b.upload(parts=videoPart, title=video\_info['messageData']['videoTitle'], tid= int(video\_info['messageData']['videoClass']), tag = tags, desc = video\_info['messageData']['videoProfile'], cover=b.cover\_up(pic\_upload\_path))

os.remove(video\_upload\_path)

return video\_callback

# 数据库更改模块

def sql\_callback(self, video\_callback, video\_info):

a\_sql = "UPDATE media\_video SET video\_url = '%s', video\_status = '%s', video\_cid = '%s' WHERE id = '%s'" % (str(video\_callback['data']['aid']), 'complete', str(video\_callback['data']['bvid']), video\_info['messageData']['id'])

b\_sql = "INSERT INTO client\_video(id, video\_title, video\_profile, video\_url, video\_cid, video\_date, video\_cover, video\_status, video\_author, create\_at, create\_by, update\_at, update\_by, is\_del) \

VALUES ('%s', '%s', '%s', '%s', '%s', '%s', '%s', '%s', '%s', '%s', '%s', '%s', '%s', %s)" % \

(video\_info['messageId'], video\_info['messageData']['videoTitle'], video\_info['messageData']['videoProfile'], str(video\_callback['data']['aid']), str(video\_callback['data']['bvid']),

video\_info['messageData']['updateAt'], video\_info['messageData']['videoPic'], 'draft', video\_info['messageData']['videoAuthor'], video\_info['messageData']['createAt'],

video\_info['messageData']['createBy'], video\_info['messageData']['updateAt'], video\_info['messageData']['updateBy'], False)

migo\_db = MySQLdb.connect("localhost","root","b66a33eca6314e6a","miguo\_matrix")

# 这个是拿来本地测试的

# migo\_db = MySQLdb.connect("localhost","root","123456","miguo\_matrix")

cursor = migo\_db.cursor()

try:

cursor.execute(a\_sql)

cursor.execute(b\_sql)

migo\_db.commit()

except:

migo\_db.rollback()

cursor.close()

migo\_db.close()

if \_\_name\_\_ == '\_\_main\_\_':

migo = MIGO()

migo.run()

数据综控后端代码

package com.miguo.matrix.entity.staff;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.EqualsAndHashCode;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import lombok.Data;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Table;

/\*\*

\* 功能描述：员工账户信息

\*

\* @author Hocassian

\*/

@EqualsAndHashCode(callSuper = true)

@Data

@Slf4j

@Entity

@Accessors(chain = true)

@ApiModel("员工实体")

@Table(name = "com\_staff")

public class Account extends BaseEntity {

@Column(name = "password")

@ApiModelProperty("密码")

private String password;

@Column(name = "name")

@ApiModelProperty("姓名")

private String name;

@Column(name = "nickname")

@ApiModelProperty("昵称")

private String nickname;

@Column(name = "level")

@ApiModelProperty("权限等级")

private String level;

}

package com.miguo.matrix.repository.staff;

import com.miguo.matrix.entity.staff.Account;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Modifying;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import java.util.Date;

/\*\*

\* @author Noah

\*/

@Repository

public interface AccountRepository extends JpaRepository<Account, String> {

/\*\*

\* 通过id(账号)查找某一个用户，忽略是否被软删除过

\* @param id

\* @return

\*/

Account findNicknameById(String id);

/\*\*

\* 通过nickname(账号)查找某一个用户，忽略是否被软删除过

\* @param nickname

\* @return

\*/

Account findByNickname(String nickname);

/\*\*

\* 软删除某一个用户

\* @param id

\* @param date

\* @param updateBy

\* @param active

\*/

@Modifying

@Transactional(rollbackFor = Exception.class)

@Query(value = "update com\_staff set is\_del = :#{#active} , update\_at = :#{#date} , update\_by = :#{#updateBy} where id = :#{#id} ", nativeQuery = true)

void deleteByNickname(Boolean active, String id, Date date, String updateBy);

/\*\*

\* 查询所有未被软删除的用户

\* @param pageable

\* @return

\*/

@Query(value = "select \* from com\_staff where is\_del = false",nativeQuery = true)

Page<Account> findAllExistAccount(Pageable pageable);

/\*\*

\* 查询所有已被软删除的用户

\* @param pageable

\* @return

\*/

@Query(value="select \* from com\_staff where is\_del = true",nativeQuery = true)

Page<Account> findAllDeletedAccount(Pageable pageable);

/\*\*

\* 根据关键词、存在状态查询

\* @param keywords

\* @param active

\* @param pageable

\* @return

\*/

@Query(value = "select \* from com\_staff WHERE ( name LIKE %:#{#keywords}% OR nickname LIKE %:#{#keywords}% ) AND is\_del = :active ", nativeQuery = true)

Page<Account> findAccountByKeywords(String keywords, Boolean active, Pageable pageable);

}

package com.miguo.matrix.service.staff;

import com.miguo.matrix.entity.staff.Account;

import com.miguo.matrix.repository.staff.AccountRepository;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import javax.servlet.http.HttpSession;

import java.util.Date;

import java.util.List;

/\*\*

\* @author Noah

\* @功能 对工作人员的管理，增删查改等功能

\*/

@Service

public class AccountService {

@Autowired

private AccountRepository accountRepository;

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

private HttpSession session;

/\*\*

\* 查找未被删除的

\* @param page

\* @param size

\* @return

\*/

public Page<Account> findAllExist(int page, int size) {

page--;

Pageable pageable = PageRequest.of(page, size);

return accountRepository.findAllExistAccount(pageable);

}

/\*\*

\* 查找已被软删除的

\* @param page

\* @param size

\* @return

\*/

public Page<Account> findAllDeleted(int page,int size){

page--;

Pageable pageable=PageRequest.of(page,size);

return accountRepository.findAllDeletedAccount(pageable);

}

/\*\*

\* 分页返回标题或内容包含某关键字的视频条目（员工使用），当关键字为空时按更新时间顺序返回所有

\* @param keywords

\* @param page

\* @param size

\* @param direction

\* @return

\*/

public Page<Account> findAllByKeywords(String keywords, String active, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "update\_at");

Boolean activeTemp = Boolean.parseBoolean(active);

return accountRepository.findAccountByKeywords(keywords, activeTemp, pageable);

}

/\*\*

\* 通过id查找某一个用户

\* @param id

\* @return

\*/

public Account findOneById(String id) {

return accountRepository.findNicknameById(id);

}

/\*\*

\* 通过nickname查找某一个用户

\* @param nickname

\* @return

\*/

public Account findOne(String nickname) {

return accountRepository.findByNickname(nickname);

}

/\*\*

\* 添加用户

\* @param account

\*/

public void add(Account account) {

account.setCreateAt(new Date());

account.setUpdateAt(new Date());

account.setPassword("123456");

account.setId(snowflakeIdWorker.nextId());

account.setCreateBy((String) session.getAttribute("user"));

account.setUpdateBy((String) session.getAttribute("user"));

account.setIsDel(false);

accountRepository.save(account);

}

/\*\*

\* 软删除或恢复用户

\* @param ids

\*/

public void modify(String ids, String active) {

String deleteNicknames = ids.split(",");

Boolean activeTemp = Boolean.parseBoolean(active);

for (String deleteNickname : deleteNicknames) {

accountRepository.deleteByNickname(activeTemp, deleteNickname, new Date(), (String) session.getAttribute("user"));

}

}

/\*\*

\* 更新用户

\* @param account

\*/

public void update(Account account) {

Account accountTemp = this.findOneById(account.getId());

BeanUtils.copyProperties(account, accountTemp);

accountTemp.setUpdateAt(new Date());

accountTemp.setUpdateBy((String) session.getAttribute("user"));

accountRepository.saveAndFlush(accountTemp);

}

/\*\*

\* 判断用户是否存在，用户不存在，直接返回，用户存在，进入下一步密码判断

\* @param nickname

\* @return

\*/

public boolean isExistByNickname(String nickname) {

Account account = accountRepository.findByNickname(nickname);

if(account != null){

return !account.getIsDel();

}

return false;

}

}

package com.miguo.matrix.entity.miniprogram;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.EqualsAndHashCode;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import lombok.Data;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Table;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 10:13

\*/

@EqualsAndHashCode(callSuper = true)

@Data

@Slf4j

@Entity

@Accessors(chain = true)

@ApiModel("小程序活动实体")

@Table(name = "vote\_activities")

public class Activity extends BaseEntity {

@Column(name = "activity\_name")

@ApiModelProperty("活动名称")

private String activityName;

@Column(name = "activity\_profile")

@ApiModelProperty("活动简介")

private String activityProfile;

@Column(name = "activity\_active")

@ApiModelProperty("活动是否活跃")

private Boolean activityActive;

@Column(name = "activity\_live")

@ApiModelProperty("活动直播链接")

private String activityLive;

}

package com.miguo.matrix.controller;

import com.miguo.matrix.dto.PageResult;

import com.miguo.matrix.dto.Result;

import com.miguo.matrix.dto.AdminSearchDto;

import com.miguo.matrix.entity.client.Article;

import com.miguo.matrix.entity.client.Swiper;

import com.miguo.matrix.entity.client.Video;

import com.miguo.matrix.entity.media.MediaVideo;

import com.miguo.matrix.entity.staff.Account;

import com.miguo.matrix.service.client.ArticleService;

import com.miguo.matrix.service.client.SwiperService;

import com.miguo.matrix.service.client.VideoService;

import com.miguo.matrix.service.media.MediaVideoService;

import com.miguo.matrix.service.staff.AccountService;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import lombok.extern.slf4j.Slf4j;

import org.springframework.amqp.rabbit.core.RabbitTemplate;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.\*;

import java.util.Date;

import java.util.HashMap;

import java.util.Map;

/\*\*

\* @author Noah

\*/

@Api("管理员的接口，该接口管理员工的增删查改")

@Slf4j

@RestController

@RequestMapping("/admin")

public class AdminController {

@Autowired

private AccountService accountService;

@Autowired

private ArticleService articleService;

@Autowired

private VideoService videoService;

@Autowired

private SwiperService swiperService;

@Autowired

private MediaVideoService mediaVideoService;

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

RabbitTemplate rabbitTemplate;

// -----------以下为员工信息的增删改查----------

@ApiOperation(value = "分页查询所有被删除的员工")

@GetMapping("/fina\_all\_deleted/{page}/{size}")

public Result<PageResult<Account>> findAllDeleted(@PathVariable("page") int page, @PathVariable("size") int size) {

Result<PageResult<Account>> result = new Result<>();

try {

Page<Account> accountPage = accountService.findAllDeleted(page, size);

// 页数减1已经在service层减了

PageResult<Account> pageResult = new PageResult<>();

pageResult.setData(accountPage.getContent());

pageResult.setPage(page).setSize(size).setTotal(accountPage.getTotalElements());

result.setData(pageResult).setMessage("success").setCode(HttpStatus.OK);

} catch (Exception e) {

result.setData(null).setMessage("fail").setCode(HttpStatus.OK);

}

return result;

}

@ApiOperation(value = "分页查询所有未被删除的员工")

@GetMapping("/fina\_all\_exist/{page}/{size}")

public Result<PageResult<Account>> findAllExist(@PathVariable("page") int page, @PathVariable("size") int size) {

Result<PageResult<Account>> result = new Result<>();

try {

Page<Account> accountPage = accountService.findAllExist(page, size);

// 页数减1已经在service层减了

PageResult<Account> pageResult = new PageResult<>();

pageResult.setData(accountPage.getContent());

pageResult.setPage(page).setSize(size).setTotal(accountPage.getTotalElements());

result.setData(pageResult).setMessage("success").setCode(HttpStatus.OK);

} catch (Exception e) {

result.setData(null).setMessage("fail").setCode(HttpStatus.OK);

}

return result;

}

@ApiOperation("分页查找所有模糊查询的员工，接收按时间正逆排序、是否下架")

@PostMapping("/find\_all\_by\_keywords")

public Result<PageResult<Account>> findAll(@RequestBody AdminSearchDto searchDto) {

Result<PageResult<Account>> result = new Result<>();

Page<Account> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = accountService.findAllByKeywords("", searchDto.getActive(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

// 当关键字为空时，查询所有

} else {

page = accountService.findAllByKeywords(searchDto.getKeywords(), searchDto.getActive(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<Account> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation(value = "通过nickname找员工信息")

@GetMapping("/find\_one\_by\_nickname/{nickname}")

public Result<Account> findOneByNickname(@PathVariable("nickname") String nickname) {

Result<Account> result = new Result<>();

try {

Account account = accountService.findOne(nickname);

result.setCode(HttpStatus.OK).setMessage("success").setData(account);

} catch (Exception e) {

result.setCode(HttpStatus.OK).setMessage("fail").setData(null);

}

return result;

}

@ApiOperation(value = "增加员工")

@PostMapping("/add")

public Result<String> add(@RequestBody Account account) {

Result<String> result = new Result<>();

try {

accountService.add(account);

result.setMessage("add").setData("success").setCode(HttpStatus.OK);

} catch (Exception e) {

result.setMessage("add").setData("fail").setCode(HttpStatus.OK);

}

return result;

}

/\*\*

\* 批量下架

\* @param ids

\* @return

\*/

@ApiOperation(value = "批量软删除员工")

@GetMapping("/modify/{ids}/{active}")

public Result<String> delete(@PathVariable("ids") String ids, @PathVariable("active") String active) {

Result<String> result = new Result<>();

result.setCode(HttpStatus.OK).setMessage("delete");

try {

accountService.modify(ids,active);

result.setData("success");

} catch (Exception e) {

result.setData("fail");

}

return result;

}

// -----------以下为文章审核的相关方法----------

@ApiOperation("分页按条件分类查找删除与否的文章")

@PostMapping("/article/find\_all\_class")

public Result<PageResult<Article>> articleFindAllClass(@RequestBody AdminSearchDto searchDto)

{

Result<PageResult<Article>> result = new Result<>();

Page<Article> pageTemp;

try {

if(searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())){

pageTemp = articleService.findAllClass("", searchDto.getPage(), searchDto.getSize(), searchDto.getActive(), searchDto.getDirection());

}

else{

pageTemp = articleService.findAllClass(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getActive(), searchDto.getDirection());

}

PageResult<Article> pageResult = new PageResult<>();

pageResult.setTotal(pageTemp.getTotalElements()).setData(pageTemp.getContent()).setPage(searchDto.getPage()).setSize(searchDto.getSize());

result.setData(pageResult).setCode(HttpStatus.OK).setMessage("success");

} catch (Exception e) {

result.setData(null).setCode(HttpStatus.OK).setMessage("fail");

}

return result;

}

@ApiOperation("分页查找所有的文章")

@PostMapping("/article/find\_all")

public Result<PageResult<Article>> articleFindAllExist(@RequestBody AdminSearchDto searchDto)

{

Result<PageResult<Article>> result = new Result<>();

Page<Article> pageTemp;

try {

if(searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())){

pageTemp = articleService.findAllExist("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

else{

pageTemp = articleService.findAllExist(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<Article> pageResult = new PageResult<>();

pageResult.setTotal(pageTemp.getTotalElements()).setData(pageTemp.getContent()).setPage(searchDto.getPage()).setSize(searchDto.getSize());

result.setData(pageResult).setCode(HttpStatus.OK).setMessage("success");

} catch (Exception e) {

result.setData(null).setCode(HttpStatus.OK).setMessage("fail");

}

return result;

}

// -----------以下为视频审核的相关方法----------

@ApiOperation("分页分类查找所有视频")

@PostMapping("/video/find\_all\_class")

public Result<PageResult<Video>> videoFindAllClass(@RequestBody AdminSearchDto searchDto){

Result<PageResult<Video>> result = new Result<>();

Page<Video> pageTemp;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

pageTemp = videoService.findAllClass("", searchDto.getPage(), searchDto.getSize(), searchDto.getActive(), searchDto.getDirection());

}

else{

pageTemp = videoService.findAllClass(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getActive(), searchDto.getDirection());

}

PageResult<Video> pageResult = new PageResult<>();

pageResult.setTotal(pageTemp.getTotalElements()).setData(pageTemp.getContent()).setPage(searchDto.getPage()).setSize(searchDto.getSize());

result.setData(pageResult).setCode(HttpStatus.OK).setMessage("success");

} catch (Exception e) {

result.setData(null).setCode(HttpStatus.OK).setMessage("fail");

}

return result;

}

@ApiOperation("分页查找所有未被删除的视频")

@PostMapping("/video/find\_all")

public Result<PageResult<Video>> videoFindAllExist(@RequestBody AdminSearchDto searchDto){

Result<PageResult<Video>> result = new Result<>();

Page<Video> pageTemp;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

pageTemp = videoService.findAllExist("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

else{

pageTemp = videoService.findAllExist(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<Video> pageResult = new PageResult<>();

pageResult.setTotal(pageTemp.getTotalElements()).setData(pageTemp.getContent()).setPage(searchDto.getPage()).setSize(searchDto.getSize());

result.setData(pageResult).setCode(HttpStatus.OK).setMessage("success");

} catch (Exception e) {

result.setData(null).setCode(HttpStatus.OK).setMessage("fail");

}

return result;

}

// -----------以下为轮播图审核的相关方法----------

@ApiOperation("分页分类查找所有视频")

@PostMapping("/swiper/find\_all\_class")

public Result<PageResult<Swiper>> swiperFindAllClass(@RequestBody AdminSearchDto searchDto){

Result<PageResult<Swiper>> result = new Result<>();

Page<Swiper> pageTemp;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

pageTemp = swiperService.findAllClass("", searchDto.getPage(), searchDto.getSize(), searchDto.getActive(), searchDto.getDirection());

}

else{

pageTemp = swiperService.findAllClass(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getActive(), searchDto.getDirection());

}

PageResult<Swiper> pageResult = new PageResult<>();

pageResult.setTotal(pageTemp.getTotalElements()).setData(pageTemp.getContent()).setPage(searchDto.getPage()).setSize(searchDto.getSize());

result.setData(pageResult).setCode(HttpStatus.OK).setMessage("success");

} catch (Exception e) {

result.setData(null).setCode(HttpStatus.OK).setMessage("fail");

}

return result;

}

@ApiOperation("分页查找所有未被删除的视频")

@PostMapping("/swiper/find\_all")

public Result<PageResult<Swiper>> swiperFindAllExist(@RequestBody AdminSearchDto searchDto){

Result<PageResult<Swiper>> result = new Result<>();

Page<Swiper> pageTemp;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

pageTemp = swiperService.findAllExist("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

else{

pageTemp = swiperService.findAllExist(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<Swiper> pageResult = new PageResult<>();

pageResult.setTotal(pageTemp.getTotalElements()).setData(pageTemp.getContent()).setPage(searchDto.getPage()).setSize(searchDto.getSize());

result.setData(pageResult).setCode(HttpStatus.OK).setMessage("success");

} catch (Exception e) {

result.setData(null).setCode(HttpStatus.OK).setMessage("fail");

}

return result;

}

// -----------以下为视频分发审核的相关方法----------

@ApiOperation("分页分类查找所有视频")

@PostMapping("/media\_video/find\_all\_class")

public Result<PageResult<MediaVideo>> mediaVideoFindAllClass(@RequestBody AdminSearchDto searchDto){

Result<PageResult<MediaVideo>> result = new Result<>();

Page<MediaVideo> pageTemp;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

pageTemp = mediaVideoService.findAllClass("", searchDto.getPage(), searchDto.getSize(), searchDto.getActive(), searchDto.getDirection());

}

else{

pageTemp = mediaVideoService.findAllClass(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getActive(), searchDto.getDirection());

}

PageResult<MediaVideo> pageResult = new PageResult<>();

pageResult.setTotal(pageTemp.getTotalElements()).setData(pageTemp.getContent()).setPage(searchDto.getPage()).setSize(searchDto.getSize());

result.setData(pageResult).setCode(HttpStatus.OK).setMessage("success");

} catch (Exception e) {

result.setData(null).setCode(HttpStatus.OK).setMessage("fail");

}

return result;

}

@ApiOperation("分页查找所有未被删除的视频")

@PostMapping("/media\_video/find\_all")

public Result<PageResult<MediaVideo>> mediaVideoFindAllExist(@RequestBody AdminSearchDto searchDto){

Result<PageResult<MediaVideo>> result = new Result<>();

Page<MediaVideo> pageTemp;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

pageTemp = mediaVideoService.findAllExist("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

else{

pageTemp = mediaVideoService.findAllExist(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<MediaVideo> pageResult = new PageResult<>();

pageResult.setTotal(pageTemp.getTotalElements()).setData(pageTemp.getContent()).setPage(searchDto.getPage()).setSize(searchDto.getSize());

result.setData(pageResult).setCode(HttpStatus.OK).setMessage("success");

} catch (Exception e) {

result.setData(null).setCode(HttpStatus.OK).setMessage("fail");

}

return result;

}

@ApiOperation("视频的分发+更新")

@PutMapping("/media\_video/distribution")

public Result<String> mediaVideoDistribution(@RequestBody MediaVideo mediaVideo){

Result<String> result = new Result<>();

try{

String messageId = snowflakeIdWorker.nextId();

Date createTime = new Date();

Map<String, Object> map=new HashMap<>(16);

map.put("messageId",messageId);

map.put("messageData",mediaVideo);

map.put("createTime",createTime);

//将消息携带绑定键值：TestDirectRouting 发送到交换机TestDirectExchange

rabbitTemplate.convertAndSend("videoSendingExchange", "videoSendingRouting", map);

mediaVideoService.update(mediaVideo);

result.setCode(HttpStatus.OK).setMessage("update").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("update").setData("fail");

}

return result;

}

}

package com.miguo.matrix.dto;

import lombok.Data;

import org.springframework.data.domain.Sort;

/\*\*

\* @author Hocassian

\*/

@Data

public class AdminSearchDto {

private int page;

private int size;

private String keywords;

private Sort.Direction direction;

private String active;

}

package com.miguo.matrix;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

/\*\*

\* @author Hocassian

\*/

@SpringBootApplication

public class Application {

public static void main(String args) {

SpringApplication.run(Application.class, args);

}

}

package com.miguo.matrix.entity.client;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.EqualsAndHashCode;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import lombok.Data;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Table;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 11:47

\*/

@EqualsAndHashCode(callSuper = true)

@Data

@Slf4j

@Entity

@Accessors(chain = true)

@ApiModel("文章实体")

@Table(name = "client\_article")

public class Article extends BaseEntity {

@Column(name = "article")

@ApiModelProperty("正文")

private String article;

@Column(name = "title")

@ApiModelProperty("标题")

private String title;

@Column(name = "author")

@ApiModelProperty("作者")

private String author;

@Column(name = "reviewer")

@ApiModelProperty("审核人")

private String reviewer;

@Column(name = "readings")

@ApiModelProperty("阅读量")

private Integer readings;

@Column(name = "status")

@ApiModelProperty("状态")

private String status;

@Column(name = "pic")

@ApiModelProperty("文章头图")

private String pic;

}

package com.miguo.matrix.repository.client;

import com.miguo.matrix.entity.client.Article;

import com.miguo.matrix.vo.web.ArticleVo;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Modifying;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import java.util.Date;

/\*\*

\* @author Hocassian

\*/

@Repository

public interface ArticleRepository extends JpaRepository<Article, String> {

/\*\*

\* ※客户方法

\*

\* 分页返回标题或内容包含某关键字且未被下架的文章条目

\* @param keywords

\* @param pageable

\* @return

\*/

@Query(value = "select \* from client\_article WHERE ( article LIKE %:#{#keywords}% OR title LIKE %:#{#keywords}% ) and is\_del = false and status = 'reviewed'",nativeQuery = true)

Page<Article> findArticleByKeywords(String keywords, Pageable pageable);

/\*\*

\* ※客户方法

\*

\* 返回挂上首页的文章条目

\* @param pageable

\* @param keywords

\* @return

\*/

@Query("SELECT new com.miguo.matrix.vo.web.ArticleVo(" +

"a.id, " +

"SUBSTRING ( a.article, 1, 70 ) as article, " +

"a.title, " +

"a.pic, " +

"a.author, " +

"a.createAt) " +

"from Article a " +

"WHERE ( a.article LIKE %:#{#keywords}% OR a.title LIKE %:#{#keywords}% ) " +

"and a.isDel = false " +

"and a.status = 'reviewed' " +

"order by a.createAt desc")

Page<ArticleVo> findSomeArticle(String keywords, Pageable pageable);

/\*\*

\* ※员工方法：进入某一篇文章修改时使用

\*

\* 通过id找文章

\* @param id

\* @return

\*/

Article findArticleById(String id);

/\*\*

\* ※员工方法

\*

\* 分页返回标题或内容包含某关键字的文章条目，当关键字为空时按更新时间顺序返回所有

\* @param keywords

\* @param pageable

\* @return

\*/

@Query(value = "select \* from client\_article WHERE article LIKE %:#{#keywords}% OR title LIKE %:#{#keywords}%", nativeQuery = true)

Page<Article> staffFindArticleByKeywords(String keywords, Pageable pageable);

/\*\*

\* ※审核方法：该方法与网络安全有关，与业务需求无关，未来迭代版本中使用

\*

\* 通过把「is\_del」改为「true」来下架多篇文章（并非删除）

\* @param id

\* @param date

\* @param updateBy

\*/

@Modifying

@Transactional(rollbackFor = Exception.class)

@Query(value = "update client\_article set is\_del = true, update\_at = :#{#date}, update\_by = :#{#updateBy} where id = :#{#id}", nativeQuery = true)

void deleteSomeById(String id, Date date, String updateBy);

/\*\*

\* ※审核方法

\*

\* 分页按条件分类查找删除与否的文章

\* @param pageable

\* @param active

\* @param keywords

\* @return

\*/

@Query(value = "select \* from client\_article where ( article LIKE %:#{#keywords}% OR title LIKE %:#{#keywords}% ) and is\_del = :active and status != 'draft'", nativeQuery = true)

Page<Article> findAllClassArticle(String keywords, Pageable pageable, Boolean active);

/\*\*

\* ※审核方法

\*

\* 分页返回所有的文章条目

\* @param pageable

\* @param keywords

\* @return

\*/

@Query(value = "select \* from client\_article where ( article LIKE %:#{#keywords}% OR title LIKE %:#{#keywords}% ) and status != 'draft'", nativeQuery = true)

Page<Article> findAllExistArticle(String keywords, Pageable pageable);

}

package com.miguo.matrix.service.client;

import com.miguo.matrix.entity.client.Article;

import com.miguo.matrix.repository.client.ArticleRepository;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import com.miguo.matrix.vo.web.ArticleVo;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import javax.servlet.http.HttpSession;

import java.util.Date;

/\*\*

\* @author Hocassian

\*/

@Service

public class ArticleService {

@Autowired

private ArticleRepository articleRepository;

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

private HttpSession session;

public ArticleService() {

}

/\*\*

\* ※员工方法

\*

\* 添加文章

\* @param article

\* @return

\*/

public String add(Article article){

article.setCreateAt(new Date());

article.setUpdateAt(new Date());

article.setId(snowflakeIdWorker.nextId());

article.setIsDel(false);

article.setCreateBy((String) session.getAttribute("user"));

article.setUpdateBy((String) session.getAttribute("user"));

articleRepository.save(article);

return article.getId();

}

/\*\*

\* ※审核方法

\*

\* 批量下架文章

\* @param ids

\*/

public void delete(String ids){

String deleteIds = ids.split(",");

for (String deleteId : deleteIds) {

articleRepository.deleteSomeById(deleteId, new Date(), (String) session.getAttribute("user"));

}

}

/\*\*

\* ※员工方法

\*

\* 删除一篇文章

\* @param id

\*/

public void deleteOne(String id){

articleRepository.deleteById(id);

}

/\*\*

\* ※客户方法

\*

\* 返回挂上首页的文章条目

\* @param keywords

\* @param page

\* @param size

\* @return

\*/

public Page<ArticleVo> findSome(String keywords, int page, int size){

page--;

Pageable pageable = PageRequest.of(page, size);

return articleRepository.findSomeArticle(keywords, pageable);

}

/\*\*

\* ※客户方法

\*

\* 分页返回标题或内容包含某关键字且未被下架的文章条目

\* @param keywords

\* @param page

\* @param size

\* @return

\*/

public Page<Article> findAllByKeywords(String keywords,int page,int size){

page--;

Pageable pageable = PageRequest.of(page, size);

return articleRepository.findArticleByKeywords(keywords,pageable);

}

/\*\*

\* ※员工方法

\*

\* 分页返回标题或内容包含某关键字的文章条目，当关键字为空时按更新时间顺序返回所有

\* @param keywords

\* @param page

\* @param size

\* @param direction

\* @return

\*/

public Page<Article> staffFindAllByKeywords(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "update\_at");

return articleRepository.staffFindArticleByKeywords(keywords, pageable);

}

/\*\*

\* ※审核方法

\*

\* 分页分类返回所有文章条目

\* @param page

\* @param size

\* @return

\*/

public Page<Article> findAllClass(String keywords, int page,int size, String active, Sort.Direction direction){

page--;

Pageable pageable=PageRequest.of(page,size, direction, "create\_at");

Boolean activeTemp = Boolean.parseBoolean(active);

return articleRepository.findAllClassArticle(keywords, pageable, activeTemp);

}

/\*\*

\* ※审核方法

\*

\* 分页不分类返回所有文章条目

\* @param page

\* @param size

\* @return

\*/

public Page<Article> findAllExist(String keywords, int page,int size, Sort.Direction direction){

page--;

Pageable pageable=PageRequest.of(page,size, direction, "create\_at");

return articleRepository.findAllExistArticle(keywords, pageable);

}

/\*\*

\* 通过id找文章

\* @param id

\* @return

\*/

public Article findOneById(String id)

{

return articleRepository.findArticleById(id);

}

/\*\*

\* 更新文章

\* @param article

\*/

public void update(Article article){

Article articleTemp=this.findOneById(article.getId());

BeanUtils.copyProperties(article, articleTemp);

articleTemp.setUpdateBy((String) session.getAttribute("user"));

articleTemp.setUpdateAt(new Date());

articleRepository.saveAndFlush(articleTemp);

}

}

package com.miguo.matrix.vo.web;

import com.fasterxml.jackson.annotation.JsonFormat;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.EqualsAndHashCode;

import lombok.NoArgsConstructor;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Table;

import java.util.Date;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 11:47

\*/

@Data

@Slf4j

@NoArgsConstructor

@AllArgsConstructor

@Accessors(chain = true)

@ApiModel("文章Vo实体")

public class ArticleVo{

private String id;

private String article;

private String title;

private String pic;

private String author;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date createAt;

}

package com.miguo.matrix.entity;

import com.fasterxml.jackson.annotation.JsonFormat;

import io.swagger.annotations.ApiModelProperty;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import javax.persistence.Column;

import javax.persistence.Id;

import javax.persistence.MappedSuperclass;

import javax.validation.constraints.NotNull;

import java.io.Serializable;

import java.util.Date;

/\*\*

\* @author Hocassian

\*/

@Data

@NoArgsConstructor

@AllArgsConstructor

@Accessors(chain = true)

@MappedSuperclass

public abstract class BaseEntity implements Serializable {

@Id

private String id;

@ApiModelProperty("创建人员")

@Column(name = "create\_by")

private String createBy;

@ApiModelProperty("更新人员")

@Column(name = "update\_by")

private String updateBy;

@ApiModelProperty("创建时间")

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

@Column(name = "create\_at")

private Date createAt;

@ApiModelProperty("更新时间")

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

@Column(name = "update\_at")

private Date updateAt;

@ApiModelProperty("是否删除")

@Column(name = "is\_del")

private Boolean isDel = Boolean.FALSE;

}

package com.miguo.matrix.vo;

import com.fasterxml.jackson.annotation.JsonFormat;

import io.swagger.annotations.ApiModel;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import java.io.Serializable;

import java.util.Date;

/\*\*

\* @author Hocassian

\*/

@Data

@NoArgsConstructor

@AllArgsConstructor

@ApiModel

@Slf4j

@Accessors(chain = true)

public abstract class BaseVo implements Serializable {

private String id;

private String createBy;

private String updateBy;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date createAt;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date updateAt;

private Boolean isDel = Boolean.FALSE;

}

package com.miguo.matrix.controller;

import com.miguo.matrix.dto.SearchDto;

import com.miguo.matrix.dto.PageResult;

import com.miguo.matrix.dto.Result;

import com.miguo.matrix.entity.client.Article;

import com.miguo.matrix.entity.client.Swiper;

import com.miguo.matrix.entity.client.Video;

import com.miguo.matrix.service.client.ArticleService;

import com.miguo.matrix.service.client.SwiperService;

import com.miguo.matrix.service.client.VideoService;

import com.miguo.matrix.vo.web.ArticleVo;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

/\*\*

\* @author Hocassian

\*/

@Api("用户专用接口，无需登陆，对视频、文章的查看")

@Slf4j

@RestController

@RequestMapping("/client")

public class ClientController {

@Autowired

private ArticleService articleService;

@Autowired

private VideoService videoService;

@Autowired

private SwiperService swiperService;

@ApiOperation("查找挂上首页的文章")

@PostMapping("/article/find\_some")

public Result<PageResult<ArticleVo>> articleFindSome(@RequestBody SearchDto searchDto) {

Result<PageResult<ArticleVo>> result = new Result<>();

Page<ArticleVo> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = articleService.findSome("", searchDto.getPage(), searchDto.getSize());

// 当关键字为空时，查询所有

} else {

page = articleService.findSome(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize());

}

PageResult<ArticleVo> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("分页查找所有模糊查询且未被软删除的文章")

@PostMapping("/article/find\_all\_by\_keywords")

public Result<PageResult<Article>> articleFindAllByKeywords(@RequestBody SearchDto searchDto) {

Result<PageResult<Article>> result = new Result<>();

Page<Article> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = articleService.findAllByKeywords("", searchDto.getPage(), searchDto.getSize());

// 当关键字为空时，查询所有

} else {

page = articleService.findAllByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize());

}

PageResult<Article> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("找某一篇文章")

@GetMapping("/article/find\_one\_by\_id/{id}")

public Result<Article> articleFindOneById(@PathVariable("id") String id) {

Result<Article> result = new Result<>();

result.setMessage("find\_one\_by\_id").setCode(HttpStatus.OK);

try {

Article article = articleService.findOneById(id);

result.setData(article).setMessage("success").setCode(HttpStatus.OK);

} catch (Exception e) {

result.setData(null).setMessage("fail").setCode(HttpStatus.OK);

}

return result;

}

@ApiOperation("分页查找所有模糊查询且未被软删除的视频")

@PostMapping("/video/find\_all\_by\_keywords")

public Result<PageResult<Video>> videoFindAllByKeywords(@RequestBody SearchDto searchDto) {

Result<PageResult<Video>> result = new Result<>();

Page<Video> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = videoService.findAllByKeywords("", searchDto.getPage(), searchDto.getSize());

// 当关键字为空时，查询所有

} else {

page = videoService.findAllByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize());

}

PageResult<Video> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("找某一支视频")

@GetMapping("/video/find\_one\_by\_id/{id}")

public Result<Video> videoFindOneById(@PathVariable("id") String id) {

Result<Video> result = new Result<>();

result.setMessage("find\_one\_by\_id").setCode(HttpStatus.OK);

try {

Video video = videoService.findOneById(id);

result.setData(video).setMessage("success").setCode(HttpStatus.OK);

} catch (Exception e) {

result.setData(null).setMessage("fail").setCode(HttpStatus.OK);

}

return result;

}

@ApiOperation("找某一个轮播图")

@GetMapping("/swiper/find\_one\_by\_id/{id}")

public Result<Swiper> swiperFindOneById(@PathVariable("id") String id) {

Result<Swiper> result = new Result<>();

result.setMessage("find\_one\_by\_id").setCode(HttpStatus.OK);

try {

Swiper swiper = swiperService.findOneById(id).orElse(null);

result.setData(swiper).setMessage("success").setCode(HttpStatus.OK);

} catch (Exception e) {

result.setData(null).setMessage("fail").setCode(HttpStatus.OK);

}

return result;

}

@ApiOperation("查找所有未被软删除的轮播图")

@PostMapping("/swiper/find\_all\_exist")

public Result<List<Swiper>> swiperFindAllByKeywords(String a) {

Result<List<Swiper>> result = new Result<>();

List<Swiper> list;

try {

list = swiperService.clientFindAllExist();

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

}

package com.miguo.matrix.controller.miniprogram;

import com.fasterxml.jackson.databind.ObjectMapper;

import com.miguo.matrix.dto.OpenIdJson;

import com.miguo.matrix.dto.Result;

import com.miguo.matrix.entity.miniprogram.\*;

import com.miguo.matrix.service.miniprogram.\*;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.\*;

import org.springframework.web.client.RestTemplate;

import java.util.List;

import java.util.Optional;

/\*\*

\* @author Hocassian

\*/

@Api("小程序专用接口，无需登陆")

@Slf4j

@RestController

@RequestMapping("/client\_miniprogram")

public class ClientMiniProgramController {

/\*

\* 需要实现：

\* 1.获取当前投票活动

\* 2.获取当前活动的投票对象

\* 3.记录每一条投票记录（增加）

\* 4.获取投票记录条目数

\* 5.获取对应的轮播图、赞助商等信息

\*

\* 注意：我目前的逻辑是，先让小程序前端随便发个值过来，返回一个目前激活的活动id，然后小程序保存这个id为全局变量，每次都返回这个id来获取其他的参数。

\* 实际上甚至可以把这些全部放在后端来完成，前端随便返回什么值，后端都按照当前激活的活动来返回对应的对象。

\* 总之无伤大雅了，运行效率都差不多。

\* \*/

private static final String IS\_VOTED = "0";

@Autowired

private MpSwiperService swiperService;

@Autowired

private MpActivityService activityService;

@Autowired

private MpGroupService groupService;

@Autowired

private MpRecordService recordService;

@Autowired

private MpMerchantService merchantService;

@Autowired

private MpNoteService noteService;

@ApiOperation("查找活躍的活動")

@PostMapping("/activity/find\_active")

public Result<List<Activity>> findActiveActivity(String a) {

Result<List<Activity>> result = new Result<>();

List<Activity> list;

try {

list = activityService.findActiveOne();

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("查找活躍的投票对象")

@GetMapping("/group/find\_active")

public Result<List<Group>> findActiveGroup(String id) {

Result<List<Group>> result = new Result<>();

List<Group> list;

try {

list = groupService.findActiveOne(id);

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("用于返回该表联结的值")

@GetMapping("/group/find\_group\_id")

public Result<Optional<Group>> findGroupById(String id){

Optional<Group> list= groupService.findById(id);

Result<Optional<Group>> result =new Result<>();

result.setCode(HttpStatus.OK).setData(list).setMessage("success");

return result;

}

@ApiOperation("查找活躍的投票对象")

@GetMapping("/group/find\_rank")

public Result<List<Group>> findRankGroup(String id) {

Result<List<Group>> result = new Result<>();

List<Group> list;

try {

list = groupService.findRank(id);

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("查找活躍的轮播图")

@GetMapping("/swiper/find\_active")

public Result<List<MpSwiper>> findActiveSwiper(String id) {

Result<List<MpSwiper>> result = new Result<>();

List<MpSwiper> list;

try {

list = swiperService.findActiveOne(id);

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("查找活躍的赞助商")

@GetMapping("/merchant/find\_active")

public Result<List<Merchant>> findActiveMerchant(String id) {

Result<List<Merchant>> result = new Result<>();

List<Merchant> list;

try {

list = merchantService.findActiveOne(id);

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("根据时间查找活躍的公告")

@GetMapping("/note/find\_active")

public Result<List<Note>> findActiveNote(String id) {

Result<List<Note>> result = new Result<>();

List<Note> list;

try {

list = noteService.findActiveOne(id);

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("为投票对象投票，传回对象id和投票人的openid，通过接口综合比对，杜绝刷票行为")

@PostMapping("/record/add")

public Result<String> voteGroup(@RequestBody Record record) {

Result<String> result = new Result<>();

result.setCode(HttpStatus.OK).setMessage("vote");

try {

String isVoted = recordService.checkIsVoted(record);

if(IS\_VOTED.equals(isVoted)){

recordService.add(record);

groupService.vote(record.getGroupId());

result.setData("success");

}

else{

result.setData("fail");

}

} catch (Exception e) {

result.setData("fail");

}

return result;

}

@PostMapping("/user/login")

public Result<OpenIdJson> userLogin(@RequestParam("code") String code) {

Result<OpenIdJson> result = new Result<>();

RestTemplate restTemplate = new RestTemplate();

try{

String appSecret = "a97bc6a54fb724206c72559a79fdda34";

String appId = "wx9e73f89a97d87442";

String resultTemp = "https://api.weixin.qq.com/sns/jscode2session?appid="

+ appId + "&secret="

+ appSecret + "&js\_code="

+ code

+ "&grant\_type=authorization\_code";

String loginResult = null;

try {

loginResult = restTemplate.getForObject(resultTemp, String.class);

}

catch (Exception e) {

e.printStackTrace();

}

ObjectMapper mapper = new ObjectMapper();

System.out.println(loginResult);

OpenIdJson openIdJson = mapper.readValue(loginResult, OpenIdJson.class);

result.setCode(HttpStatus.OK).setMessage("code").setData(openIdJson);

}

catch (Exception e) {

e.printStackTrace();

result.setCode(HttpStatus.OK).setMessage("fail").setData(null);

}

return result;

}

}

package com.miguo.matrix.service.staff;

import com.miguo.matrix.vo.CountVo;

import com.miguo.matrix.repository.staff.QualityRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import java.util.List;

import java.util.Map;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2020-03-25 14:57

\*/

@Service

public class CountService {

@Autowired

private QualityRepository qualityRepository;

public Map<String, Integer> findAllCount(){

return qualityRepository.countAll();

// return null;

}

}

package com.miguo.matrix.vo;

import io.swagger.annotations.ApiModel;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import javax.persistence.Entity;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2020-03-25 14:35

\*/

@Data

@Slf4j

@NoArgsConstructor

@AllArgsConstructor

@Accessors(chain = true)

@ApiModel("数据统计实体")

public class CountVo {

/\*\*

\* 文章数

\*/

private Integer articleWebCount;

/\*\*

\* 视频数

\*/

private Integer videoWebCount;

/\*\*

\* 轮播图数

\*/

private Integer swiperWebCount;

/\*\*

\* 网站浏览量（暂无）

\*/

private Integer viewWebCount;

/\*\*

\* 使用次数

\*/

private Integer activityMiniCount;

/\*\*

\* 小程序使用人数

\*/

private Integer userMiniCount;

/\*\*

\* 赞助商数

\*/

private Integer merchantMiniCount;

/\*\*

\* 员工数

\*/

private Integer staffWebCount;

}

package com.miguo.matrix.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.web.cors.CorsConfiguration;

import org.springframework.web.cors.UrlBasedCorsConfigurationSource;

import org.springframework.web.filter.CorsFilter;

/\*\*

\* @author Hocassian

\*/

@Configuration

public class CrosConfig {

private CorsConfiguration buildConfig() {

CorsConfiguration corsConfiguration = new CorsConfiguration();

corsConfiguration.addAllowedOrigin("\*");

corsConfiguration.addAllowedHeader("\*");

corsConfiguration.addAllowedMethod("\*");

corsConfiguration.setAllowCredentials(true);

return corsConfiguration;

}

@Bean

public CorsFilter corsFilter() {

UrlBasedCorsConfigurationSource source = new UrlBasedCorsConfigurationSource();

source.registerCorsConfiguration("/\*\*", buildConfig());

return new CorsFilter(source);

}

}

package com.miguo.matrix.utils;

import com.fasterxml.jackson.core.JsonGenerator;

import com.fasterxml.jackson.databind.JsonSerializer;

import com.fasterxml.jackson.databind.SerializerProvider;

import org.springframework.http.HttpStatus;

import java.io.IOException;

/\*\*

\* @author Hocassian

\*/

public class EnumSerializer extends JsonSerializer {

@Override

public void serialize(Object object, JsonGenerator jsonGenerator, SerializerProvider serializerProvider) throws IOException {

if (object instanceof HttpStatus) {

HttpStatus httpStatus = (HttpStatus) object;

jsonGenerator.writeNumber(httpStatus.value());

} else {

jsonGenerator.writeString(object.toString());

}

}

}

package com.miguo.matrix.entity.miniprogram;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.EqualsAndHashCode;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import lombok.Data;

import javax.persistence.\*;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 11:41

\*/

@EqualsAndHashCode(callSuper = true)

@Data

@Slf4j

@Entity

@Accessors(chain = true)

@ApiModel("小程序投票对象实体")

@Table(name = "vote\_groups")

public class Group extends BaseEntity {

@Column(name = "activity\_id")

@ApiModelProperty("活动id")

private String activityId;

@Column(name = "group\_name")

@ApiModelProperty("组名")

private String groupName;

@Column(name = "group\_profile")

@ApiModelProperty("简介")

private String groupProfile;

@Column(name = "group\_votes")

@ApiModelProperty("获票数")

private Integer groupVotes;

@Column(name = "group\_pic")

@ApiModelProperty("缩略图")

private String groupPic;

@Column(name = "group\_pic\_hd")

@ApiModelProperty("高清图")

private String groupPicHd;

}

package com.miguo.matrix.vo.miniprogram;

import com.fasterxml.jackson.annotation.JsonFormat;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import java.util.Date;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 11:41

\*/

@Data

@Slf4j

@NoArgsConstructor

@AllArgsConstructor

@Accessors(chain = true)

@ApiModel("小程序投票对象Vo实体")

public class GroupVo {

private String id;

private String createBy;

private String updateBy;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date createAt;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date updateAt;

private Boolean isDel = Boolean.FALSE;

@ApiModelProperty("活动id")

private String activityId;

@ApiModelProperty("组名")

private String groupName;

@ApiModelProperty("简介")

private String groupProfile;

@ApiModelProperty("获票数")

private Integer groupVotes;

@ApiModelProperty("缩略图")

private String groupPic;

@ApiModelProperty("高清图")

private String groupPicHd;

@ApiModelProperty("活动名称")

private String activityName;

}

package com.miguo.matrix.controller;

import com.miguo.matrix.dto.Result;

import com.miguo.matrix.entity.staff.Account;

import com.miguo.matrix.service.staff.AccountService;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.Enumeration;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-27 17:04

\*/

@Api("登陆接口")

@Slf4j

@RestController

@RequestMapping("/account")

public class LoginController {

@Autowired

private AccountService accountService;

@Autowired

private HttpSession session;

private static final String STAFF = "2";

private static final String ADMIN = "1";

/\*\*

\* 系统登录

\*/

@ApiOperation(value = "系统登录")

@PostMapping("/login")

public Result<String> login(@RequestBody Account account, HttpServletRequest request) {

Account accountTemp = accountService.findOne(account.getNickname());

Result<String> result = new Result<>();

// HttpSession session = request.getSession;

if (!accountService.isExistByNickname(account.getNickname())) {

result.setData("fail").setMessage("用户不存在！").setCode(HttpStatus.OK);

} else {

if (account.getPassword().equals((accountTemp.getPassword()))) {

// 管理员登陆

if(accountTemp.getLevel().equals(ADMIN)){

result.setData("admin").setMessage("登陆成功！").setCode(HttpStatus.OK);

// 200分钟有效期

session.setMaxInactiveInterval(1000 \* 60 \* 200);

request.getSession().setAttribute("user", account.getNickname());

System.out.print((String) session.getAttribute("user"));

}

// 普通员工登陆

else {

result.setData("staff").setMessage("登陆成功！").setCode(HttpStatus.OK);

// 100分钟有效期

session.setMaxInactiveInterval(1000 \* 60 \* 100);

request.getSession().setAttribute("user", account.getNickname());

}

} else {

result.setData("fail").setMessage("密码错误！").setCode(HttpStatus.OK);

}

}

return result;

}

@ApiOperation(value = "系统注销")

@PostMapping("/logout")

public Result<String> logout() {

Result<String> result = new Result<>();

Enumeration em = session.getAttributeNames();

while (em.hasMoreElements()) {

session.removeAttribute(em.nextElement().toString());

}

return result.setData("logout").setMessage("已经注销！").setCode(HttpStatus.OK);

}

}

package com.miguo.matrix.controller;

import com.miguo.matrix.dto.PageResult;

import com.miguo.matrix.dto.Result;

import com.miguo.matrix.dto.SearchDto;

import com.miguo.matrix.entity.media.MediaVideo;

import com.miguo.matrix.service.media.MediaVideoService;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.\*;

import javax.servlet.http.HttpSession;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2020-03-13 10:24

\*/

@Api("员工管理接口，对视频、文章进行全平台分发")

@Slf4j

@RestController

@RequestMapping("/media")

public class MediaController {

@Autowired

private MediaVideoService videoService;

@Autowired

private HttpSession session;

@ApiOperation("视频的添加")

@PostMapping("/video/add")

public Result<String> videoAdd(@RequestBody MediaVideo mediaVideo){

Result<String> result = new Result<>();

try{

videoService.add(mediaVideo);

result.setCode(HttpStatus.OK).setMessage("add").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("add").setData("fail");

}

return result;

}

@ApiOperation("删除单个视频")

@DeleteMapping("/video/delete\_one/{id}")

public Result<String> videoDeleteOne(@PathVariable("id") String id){

Result<String> result = new Result<>();

result.setCode(HttpStatus.OK).setMessage("delete");

try{

videoService.deleteOne(id);

result.setData("success");

}catch (Exception e){

result.setData("fail");

}

return result;

}

@ApiOperation("视频的更新")

@PutMapping("/video/update")

public Result<String> videoUpdate(@RequestBody MediaVideo mediaVideo){

Result<String> result = new Result<>();

try{

videoService.update(mediaVideo);

result.setCode(HttpStatus.OK).setMessage("update").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("update").setData("fail");

}

return result;

}

@ApiOperation("分页查找所有模糊查询的视频")

@PostMapping("/video/find\_all\_by\_keywords")

public Result<PageResult<MediaVideo>> videoFindAllByKeywords(@RequestBody SearchDto searchDto) {

Result<PageResult<MediaVideo>> result = new Result<>();

Page<MediaVideo> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = videoService.staffFindAllByKeywords("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

// 当关键字为空时，查询所有

} else {

page = videoService.staffFindAllByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<MediaVideo> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("获取某一个视频")

@GetMapping("/video/find\_one\_by\_id/{id}")

public Result<MediaVideo> findVideoById(@PathVariable("id") String id){

Result<MediaVideo> result = new Result<>();

try {

MediaVideo list = videoService.findOneById(id);

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

}

package com.miguo.matrix.entity.media;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import lombok.EqualsAndHashCode;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Table;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 11:53

\*/

@EqualsAndHashCode(callSuper = true)

@Slf4j

@Data

@Entity

@Accessors(chain = true)

@ApiModel("视频实体")

@Table(name = "media\_video")

public class MediaVideo extends BaseEntity {

@Column(name = "video\_url")

@ApiModelProperty("视频链接")

private String videoUrl;

@Column(name = "video\_cid")

@ApiModelProperty("视频cid")

private String videoCid;

@Column(name = "video\_title")

@ApiModelProperty("视频标题")

private String videoTitle;

@Column(name = "video\_profile")

@ApiModelProperty("视频简介")

private String videoProfile;

@Column(name = "video\_pic")

@ApiModelProperty("视频封面")

private String videoPic;

@Column(name = "video\_author")

@ApiModelProperty("创作团队")

private String videoAuthor;

@Column(name = "video\_status")

@ApiModelProperty("状态")

private String videoStatus;

@Column(name = "video\_class")

@ApiModelProperty("视频类别")

private String videoClass;

@Column(name = "video\_tag")

@ApiModelProperty("视频标签")

private String videoTag;

@Column(name = "video\_reviewer")

@ApiModelProperty("视频审核人")

private String videoReviewer;

@Column(name = "video\_path")

@ApiModelProperty("视频本地路径")

private String videoPath;

}

package com.miguo.matrix.repository.media;

import com.miguo.matrix.entity.media.MediaVideo;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

/\*\*

\* @author Hocassian

\*/

@Repository

public interface MediaVideoRepository extends JpaRepository<MediaVideo,String> {

/\*\*

\* ※员工方法：进入某一篇视频修改时使用

\*

\* 通过id找视频

\* @param id

\* @return

\*/

MediaVideo findVideoById(String id);

/\*\*

\* ※员工方法

\*

\* 分页返回视频标题或内容包含某关键字的视频条目，当关键字为空时按更新时间顺序返回所有

\* @param keywords

\* @param pageable

\* @return

\*/

@Query(value = "select \* from media\_video WHERE video\_title LIKE %:#{#keywords}% OR video\_profile LIKE %:#{#keywords}%", nativeQuery = true)

Page<MediaVideo> staffFindVideoByKeywords(String keywords, Pageable pageable);

/\*\*

\* ※审核方法

\*

\* 分页不分类返回所有的视频条目

\* @param pageable

\* @param keywords

\* @return

\*/

@Query(value = "select \* from media\_video where ( video\_title LIKE %:#{#keywords}% OR video\_profile LIKE %:#{#keywords}% ) and video\_status != 'draft'", nativeQuery = true)

Page<MediaVideo> findAllExistVideo(String keywords, Pageable pageable);

/\*\*

\* ※审核方法

\*

\* 分页分类返回所有的视频条目

\* @param pageable

\* @param active

\* @param keywords

\* @return

\*/

@Query(value = "select \* from media\_video where ( video\_title LIKE %:#{#keywords}% OR video\_profile LIKE %:#{#keywords}% ) and is\_del = :active and video\_status != 'draft'",nativeQuery = true)

Page<MediaVideo> findAllClassVideo(String keywords, Pageable pageable, Boolean active);

}

package com.miguo.matrix.service.media;

import com.miguo.matrix.entity.media.MediaVideo;

import com.miguo.matrix.repository.media.MediaVideoRepository;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import javax.servlet.http.HttpSession;

import java.util.Date;

/\*\*

\* @author Hocassian

\*/

@Service

public class MediaVideoService {

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

private MediaVideoRepository videoRepository;

@Autowired

private HttpSession session;

/\*\*

\* 添加视频

\* @param mediaVideo

\*/

public void add(MediaVideo mediaVideo){

mediaVideo.setCreateAt(new Date());

mediaVideo.setUpdateAt(new Date());

mediaVideo.setUpdateBy((String) session.getAttribute("user"));

mediaVideo.setCreateBy((String) session.getAttribute("user"));

mediaVideo.setId(snowflakeIdWorker.nextId());

mediaVideo.setIsDel(false);

videoRepository.save(mediaVideo);

}

/\*\*

\* 分页返回标题或内容包含某关键字的视频条目（员工使用），当关键字为空时按更新时间顺序返回所有

\* @param keywords

\* @param page

\* @param size

\* @param direction

\* @return

\*/

public Page<MediaVideo> staffFindAllByKeywords(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "update\_at");

return videoRepository.staffFindVideoByKeywords(keywords,pageable);

}

/\*\*

\* 分页分类返回所有视频条目

\* @param page

\* @param size

\* @return

\*/

public Page<MediaVideo> findAllClass(String keywords, int page, int size, String active, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page,size, direction, "create\_at");

Boolean activeTemp = Boolean.parseBoolean(active);

return videoRepository.findAllClassVideo(keywords, pageable, activeTemp);

}

/\*\*

\* 分页不分类返回所有视频条目

\* @param page

\* @param size

\* @return

\*/

public Page<MediaVideo> findAllExist(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "create\_at");

return videoRepository.findAllExistVideo(keywords, pageable);

}

/\*\*

\* 通过id找视频

\* @param id

\* @return

\*/

public MediaVideo findOneById(String id){

return videoRepository.findVideoById(id);

}

/\*\*

\* 更新视频

\* @param mediaVideo

\*/

public void update(MediaVideo mediaVideo){

MediaVideo mediaVideoTemp =this.findOneById(mediaVideo.getId());

BeanUtils.copyProperties(mediaVideo, mediaVideoTemp);

mediaVideoTemp.setUpdateAt(new Date());

mediaVideoTemp.setUpdateBy((String) session.getAttribute("user"));

videoRepository.saveAndFlush(mediaVideoTemp);

}

/\*\*

\* 删除一个视频

\* @param id

\*/

public void deleteOne(String id){

videoRepository.deleteById(id);

}

}

package com.miguo.matrix.entity.miniprogram;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.EqualsAndHashCode;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import lombok.Data;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Table;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 10:21

\*/

@EqualsAndHashCode(callSuper = true)

@Data

@Slf4j

@Entity

@Accessors(chain = true)

@ApiModel("小程序赞助商实体")

@Table(name = "vote\_merchants")

public class Merchant extends BaseEntity {

@Column(name = "activity\_id")

@ApiModelProperty("活动id")

private String activityId;

@Column(name = "merchant\_name")

@ApiModelProperty("赞助商名称")

private String merchantName;

@Column(name = "merchant\_info")

@ApiModelProperty("赞助商信息")

private String merchantInfo;

@Column(name = "merchant\_logo")

@ApiModelProperty("赞助商图标（链接地址）")

private String merchantLogo;

}

package com.miguo.matrix.vo.miniprogram;

import com.fasterxml.jackson.annotation.JsonFormat;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import java.util.Date;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 10:21

\*/

@Data

@Slf4j

@NoArgsConstructor

@AllArgsConstructor

@Accessors(chain = true)

@ApiModel("小程序赞助商Vo实体")

public class MerchantVo {

private String id;

private String createBy;

private String updateBy;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date createAt;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date updateAt;

private Boolean isDel = Boolean.FALSE;

@ApiModelProperty("活动id")

private String activityId;

@ApiModelProperty("赞助商名称")

private String merchantName;

@ApiModelProperty("赞助商信息")

private String merchantInfo;

@ApiModelProperty("赞助商图标（链接地址）")

private String merchantLogo;

@ApiModelProperty("活动名称")

private String activityName;

}

package com.miguo.matrix.repository.miniprogram;

import com.miguo.matrix.entity.miniprogram.Activity;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import java.util.List;

/\*\*

\* @author Hocassian

\*/

@Repository

public interface MpActivityRepository extends JpaRepository<Activity,String> {

/\*\*

\* 查找所有标题或者内容包含该关键字的活动（不分页，输入活动相关信息时联结用）

\* @param keywords

\* @return

\*/

@Query(value = "select \* from vote\_activities WHERE activity\_name LIKE %:#{#keywords}% OR activity\_profile LIKE %:#{#keywords}%",nativeQuery = true)

List<Activity> findActivityByKeywordsFromInput(String keywords);

/\*\*

\* 分页查找所有标题或者内容包含该关键字且未被软删除的活动（录入活动页面用,「keywords」为空时返回所有）

\* @param keywords

\* @param pageable

\* @return

\*/

@Query(value = "select \* from vote\_activities WHERE activity\_name LIKE %:#{#keywords}% OR activity\_profile LIKE %:#{#keywords}%",nativeQuery = true)

Page<Activity> findActivityByKeywords(String keywords, Pageable pageable);

/\*\*

\* 通过id找活动

\* @param id

\* @return

\*/

Activity findActivityById(String id);

/\*\*

\* 查找被激活的活动

\* @return

\*/

@Query(value = "select \* from vote\_activities WHERE activity\_active = true",nativeQuery = true)

List<Activity> findActiveActivity();

}

package com.miguo.matrix.service.miniprogram;

import com.miguo.matrix.entity.miniprogram.Activity;

import com.miguo.matrix.repository.miniprogram.MpActivityRepository;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import javax.servlet.http.HttpSession;

import java.util.Date;

import java.util.List;

import java.util.Optional;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-12-17 10:02

\*/

@Service

public class MpActivityService {

@Autowired

private MpActivityRepository activityRepository;

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

private HttpSession session;

/\*\*

\* 添加活动

\* @param activity

\*/

public void add(Activity activity){

activity.setCreateAt(new Date());

activity.setUpdateAt(new Date());

activity.setId(snowflakeIdWorker.nextId());

activity.setIsDel(false);

activity.setCreateBy((String) session.getAttribute("user"));

activity.setUpdateBy((String) session.getAttribute("user"));

activityRepository.save(activity);

}

/\*\*

\* 删除&批量删除

\* @param list

\*/

public void delete(List<Activity> list){

activityRepository.deleteInBatch(list);

}

/\*\*

\* 分页查找所有标题或者内容包含该关键字且未被软删除的活动（录入活动页面用,「keywords」为空时返回所有）

\* @param page

\* @param size

\* @return

\*/

public Page<Activity> findAllActivityByKeywords(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "update\_at");

return activityRepository.findActivityByKeywords(keywords,pageable);

}

/\*\*

\* 查找所有标题或者内容包含该关键字的活动（不分页，输入活动相关信息时联结用）

\* @param keywords

\* @return

\*/

public List<Activity> findAllActivityByKeywordsFromInput (String keywords) {

return activityRepository.findActivityByKeywordsFromInput(keywords);

}

/\*\*

\* 通过id查找活动（用于查找）

\* @param id

\* @return

\*/

public Optional<Activity> findById(String id)

{

return activityRepository.findById(id);

}

/\*\*

\* 通过id查找活动（用于更新）

\* @param id

\* @return

\*/

public Activity findOneById(String id)

{

return activityRepository.findActivityById(id);

}

/\*\*

\* 更新活动

\* @param activity

\*/

public void update(Activity activity){

Activity activityTemp=this.findOneById(activity.getId());

BeanUtils.copyProperties(activity, activityTemp);

activityTemp.setUpdateBy((String) session.getAttribute("user"));

activityTemp.setUpdateAt(new Date());

activityRepository.saveAndFlush(activityTemp);

}

public List<Activity> findActiveOne() { return activityRepository.findActiveActivity(); }

}

package com.miguo.matrix.repository.miniprogram;

import com.miguo.matrix.entity.miniprogram.Group;

import com.miguo.matrix.vo.miniprogram.GroupVo;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Modifying;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import java.util.Date;

import java.util.List;

/\*\*

\* @author Hocassian

\*/

@Repository

public interface MpGroupRepository extends JpaRepository<Group,String> {

/\*\*

\* 分页查找所有标题或者内容包含该关键字且未被软删除的活动（录入活动页面用,「keywords」为空时返回所有）

\* @param keywords

\* @param pageable

\* @return

\*/

@Query("SELECT new com.miguo.matrix.vo.miniprogram.GroupVo(" +

"g.id," +

"g.createBy," +

"g.updateBy," +

"g.createAt," +

"g.updateAt," +

"g.isDel," +

"g.activityId," +

"g.groupName," +

"g.groupProfile," +

"g.groupVotes," +

"g.groupPic," +

"g.groupPicHd," +

"a.activityName) " +

"FROM Group g " +

"LEFT JOIN Activity a ON g.activityId=a.id WHERE g.groupName LIKE %:#{#keywords}% OR g.groupProfile LIKE %:#{#keywords}%")

Page<GroupVo> findGroupByKeywords(String keywords, Pageable pageable);

/\*\*

\* 查找所有标题或者内容包含该关键字的活动（不分页，输入活动相关信息时联结用）

\* @param keywords

\* @return

\*/

@Query(value = "select \* from vote\_groups WHERE group\_name LIKE %:#{#keywords}% OR group\_profile LIKE %:#{#keywords}%",nativeQuery = true)

List<Group> findGroupByKeywordsFromInput(String keywords);

/\*\*

\* 通过id找投票对象

\* @param id

\* @return

\*/

Group findGroupById(String id);

/\*\*

\* 通过id找激活的投票对象序列

\* @param id

\* @return

\*/

@Query(value = "select \* from vote\_groups WHERE activity\_id = :#{#id}", nativeQuery = true)

List<Group> findActiveGroup(String id);

/\*\*

\* 通过id找激活的投票对象序列（排行榜）

\* @param id

\* @return

\*/

@Query(value = "select \* from vote\_groups WHERE activity\_id = :#{#id} order by group\_votes DESC limit 3", nativeQuery = true)

List<Group> findRankGroup(String id);

/\*\*

\* 通过id定位投票对象，然后实现投票数的自增（+1）

\* @param id

\*/

@Modifying

@Transactional(rollbackFor = Exception.class)

@Query(value = "update `vote\_groups` set group\_votes=group\_votes+1 where id = :#{#id}", nativeQuery = true)

void voteForGroup(String id);

}

package com.miguo.matrix.service.miniprogram;

import com.miguo.matrix.entity.miniprogram.Group;

import com.miguo.matrix.repository.miniprogram.MpGroupRepository;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import com.miguo.matrix.vo.miniprogram.GroupVo;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import javax.servlet.http.HttpSession;

import java.util.Date;

import java.util.List;

import java.util.Optional;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-12-17 10:02

\*/

@Service

public class MpGroupService {

@Autowired

private MpGroupRepository groupRepository;

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

private HttpSession session;

/\*\*

\* 添加投票对象

\* @param groupVo

\*/

public void add(GroupVo groupVo){

Group groupTemp = new Group();

BeanUtils.copyProperties(groupVo, groupTemp);

groupTemp.setCreateAt(new Date());

groupTemp.setUpdateAt(new Date());

groupTemp.setId(snowflakeIdWorker.nextId());

groupTemp.setIsDel(false);

groupTemp.setCreateBy((String) session.getAttribute("user"));

groupTemp.setUpdateBy((String) session.getAttribute("user"));

groupTemp.setGroupVotes(0);

groupRepository.save(groupTemp);

}

/\*\*

\* 删除&批量删除

\* @param list

\*/

public void delete(List<Group> list){

groupRepository.deleteInBatch(list);

}

public Page<GroupVo> findGroupByKeywords(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "updateAt");

return groupRepository.findGroupByKeywords(keywords,pageable);

}

public List<Group> findAllActivityByKeywordsFromInput (String keywords) {

return groupRepository.findGroupByKeywordsFromInput(keywords);

}

public Optional<Group> findById(String id)

{

return groupRepository.findById(id);

}

public Group findOneById(String id)

{

return groupRepository.findGroupById(id);

}

public void update(Group group){

Group groupTemp=this.findOneById(group.getId());

BeanUtils.copyProperties(group, groupTemp);

groupTemp.setUpdateBy((String) session.getAttribute("user"));

groupTemp.setUpdateAt(new Date());

groupRepository.saveAndFlush(groupTemp);

}

public List<Group> findActiveOne(String id) {

return groupRepository.findActiveGroup(id);

}

public List<Group> findRank(String id) {

return groupRepository.findRankGroup(id);

}

public void vote(String id){

groupRepository.voteForGroup(id);

}

}

package com.miguo.matrix.repository.miniprogram;

import com.miguo.matrix.entity.miniprogram.Merchant;

import com.miguo.matrix.vo.miniprogram.MerchantVo;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import java.util.List;

/\*\*

\* @author Hocassian

\*/

@Repository

public interface MpMerchantRepository extends JpaRepository<Merchant,String> {

/\*\*

\* 分页查找所有标题或者内容包含该关键字且未被软删除的活动（录入活动页面用,「keywords」为空时返回所有）

\* @param keywords

\* @param pageable

\* @return

\*/

@Query("SELECT new com.miguo.matrix.vo.miniprogram.MerchantVo(" +

"g.id," +

"g.createBy," +

"g.updateBy," +

"g.createAt," +

"g.updateAt," +

"g.isDel," +

"g.activityId," +

"g.merchantName," +

"g.merchantInfo," +

"g.merchantLogo," +

"a.activityName) " +

"FROM Merchant g " +

"LEFT JOIN Activity a ON g.activityId=a.id WHERE g.merchantName LIKE %:#{#keywords}% OR g.merchantInfo LIKE %:#{#keywords}%")

Page<MerchantVo> findMerchantByKeywords(String keywords, Pageable pageable);

/\*\*

\* 通过id找投票对象

\* @param id

\* @return

\*/

Merchant findMerchantById(String id);

/\*\*

\* 通过id找激活的投票对象

\* @param id

\* @return

\*/

@Query(value = "select \* from vote\_merchants WHERE activity\_id = :#{#id}",nativeQuery = true)

List<Merchant> findActiveMerchant(String id);

}

package com.miguo.matrix.service.miniprogram;

import com.miguo.matrix.entity.miniprogram.Merchant;

import com.miguo.matrix.repository.miniprogram.MpMerchantRepository;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import com.miguo.matrix.vo.miniprogram.MerchantVo;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import javax.servlet.http.HttpSession;

import java.util.Date;

import java.util.List;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-12-17 10:02

\*/

@Service

public class MpMerchantService {

@Autowired

private MpMerchantRepository merchantRepository;

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

private HttpSession session;

public void add(Merchant merchant){

merchant.setCreateAt(new Date());

merchant.setUpdateAt(new Date());

merchant.setId(snowflakeIdWorker.nextId());

merchant.setIsDel(false);

merchant.setCreateBy((String) session.getAttribute("user"));

merchant.setUpdateBy((String) session.getAttribute("user"));

merchantRepository.save(merchant);

}

public void delete(List<Merchant> list){

merchantRepository.deleteInBatch(list);

}

public Page<MerchantVo> findMerchantByKeywords(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "updateAt");

return merchantRepository.findMerchantByKeywords(keywords,pageable);

}

public Merchant findOneById(String id)

{

return merchantRepository.findMerchantById(id);

}

public void update(Merchant merchant){

Merchant merchantTemp=this.findOneById(merchant.getId());

BeanUtils.copyProperties(merchant, merchantTemp);

merchantTemp.setUpdateBy((String) session.getAttribute("user"));

merchantTemp.setUpdateAt(new Date());

merchantRepository.saveAndFlush(merchantTemp);

}

public List<Merchant> findActiveOne(String id) {

return merchantRepository.findActiveMerchant(id);

}

}

package com.miguo.matrix.repository.miniprogram;

import com.miguo.matrix.entity.miniprogram.Note;

import com.miguo.matrix.vo.miniprogram.NoteVo;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import java.util.Date;

import java.util.List;

/\*\*

\* @author Hocassian

\*/

@Repository

public interface MpNoteRepository extends JpaRepository<Note,String> {

/\*\*

\* 分页查找所有标题或者内容包含该关键字且未被软删除的活动（录入活动页面用,「keywords」为空时返回所有）

\* @param keywords

\* @param pageable

\* @return

\*/

@Query("SELECT new com.miguo.matrix.vo.miniprogram.NoteVo(" +

"g.id," +

"g.createBy," +

"g.updateBy," +

"g.createAt," +

"g.updateAt," +

"g.isDel," +

"g.activityId," +

"g.noteName," +

"g.noteDisplayTime," +

"g.noteDetails," +

"a.activityName) " +

"FROM Note g " +

"LEFT JOIN Activity a ON g.activityId=a.id WHERE g.noteName LIKE %:#{#keywords}% OR g.noteDetails LIKE %:#{#keywords}%")

Page<NoteVo> findNoteByKeywords(String keywords, Pageable pageable);

/\*\*

\* 通过id找公告对象

\* @param id

\* @return

\*/

Note findNoteById(String id);

/\*\*

\* 通过id找激活的公告对象

\* @param id

\* @param date

\* @return

\*/

@Query(value = "select \* from vote\_notes WHERE activity\_id = :#{#id} and note\_display\_time < cast(:date as datetime) order by note\_display\_time DESC",nativeQuery = true)

List<Note> findActiveNote(String id, Date date);

}

package com.miguo.matrix.service.miniprogram;

import com.miguo.matrix.entity.miniprogram.Note;

import com.miguo.matrix.repository.miniprogram.MpNoteRepository;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import com.miguo.matrix.vo.miniprogram.NoteVo;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import javax.servlet.http.HttpSession;

import java.util.Date;

import java.util.List;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-12-17 10:02

\*/

@Service

public class MpNoteService {

@Autowired

private MpNoteRepository noteRepository;

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

private HttpSession session;

public void add(Note note){

note.setCreateAt(new Date());

note.setUpdateAt(new Date());

note.setId(snowflakeIdWorker.nextId());

note.setIsDel(false);

note.setCreateBy((String) session.getAttribute("user"));

note.setUpdateBy((String) session.getAttribute("user"));

noteRepository.save(note);

}

public void delete(List<Note> list){

noteRepository.deleteInBatch(list);

}

public Page<NoteVo> findNoteByKeywords(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "updateAt");

return noteRepository.findNoteByKeywords(keywords,pageable);

}

public Note findOneById(String id)

{

return noteRepository.findNoteById(id);

}

public void update(Note note){

Note noteTemp=this.findOneById(note.getId());

BeanUtils.copyProperties(note, noteTemp);

noteTemp.setUpdateBy((String) session.getAttribute("user"));

noteTemp.setUpdateAt(new Date());

noteRepository.saveAndFlush(noteTemp);

}

public List<Note> findActiveOne(String id) {

return noteRepository.findActiveNote(id,new Date());

}

}

package com.miguo.matrix.repository.miniprogram;

import com.miguo.matrix.entity.miniprogram.Record;

import com.miguo.matrix.vo.miniprogram.RecordVo;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import java.util.List;

/\*\*

\* @author Hocassian

\*/

@Repository

public interface MpRecordRepository extends JpaRepository<Record,String> {

/\*\*

\* 分页查找所有标题或者内容包含该关键字且未被软删除的活动（录入活动页面用,「keywords」为空时返回所有）

\* @param keywords

\* @param pageable

\* @return

\*/

@Query("SELECT new com.miguo.matrix.vo.miniprogram.RecordVo(" +

"r.id," +

"r.createBy," +

"r.updateBy," +

"r.createAt," +

"r.updateAt," +

"r.isDel," +

"r.activityId," +

"r.groupId," +

"r.recordNickname," +

"r.recordOpenid," +

"r.recordDate," +

"g.groupName) " +

"FROM Record r " +

"LEFT JOIN Group g ON r.groupId=g.id " +

"WHERE r.recordOpenid LIKE %:#{#keywords}%")

Page<RecordVo> findRecordByKeywords(String keywords, Pageable pageable);

/\*\*

\* 通过id找投票记录

\* @param id

\* @return

\*/

Record findRecordById(String id);

/\*\*

\* 通过openid确定这个用户投过票没

\* @param openid

\* @return

\*/

@Query(value = "SELECT case when record\_openid = :openid then 1 else 0 end FROM vote\_records", nativeQuery = true)

String findIsVoted(String openid);

}

package com.miguo.matrix.service.miniprogram;

import com.miguo.matrix.entity.miniprogram.Record;

import com.miguo.matrix.repository.miniprogram.MpRecordRepository;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import com.miguo.matrix.vo.miniprogram.RecordVo;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import javax.servlet.http.HttpSession;

import java.util.Date;

import java.util.List;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-12-17 10:02

\*/

@Service

public class MpRecordService {

@Autowired

private MpRecordRepository recordRepository;

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

private HttpSession session;

public void add(Record record){

record.setCreateAt(new Date());

record.setUpdateAt(new Date());

record.setId(snowflakeIdWorker.nextId());

record.setIsDel(false);

record.setCreateBy((String) session.getAttribute("user"));

record.setUpdateBy((String) session.getAttribute("user"));

recordRepository.save(record);

}

public void delete(List<Record> list){

recordRepository.deleteInBatch(list);

}

public Page<RecordVo> findRecordByKeywords(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "updateAt");

return recordRepository.findRecordByKeywords(keywords,pageable);

}

public Record findOneById(String id)

{

return recordRepository.findRecordById(id);

}

public void update(Record record){

Record recordTemp=this.findOneById(record.getId());

BeanUtils.copyProperties(record, recordTemp);

recordTemp.setUpdateBy((String) session.getAttribute("user"));

recordTemp.setUpdateAt(new Date());

recordRepository.saveAndFlush(recordTemp);

}

public String checkIsVoted(Record record){

String openid = record.getRecordOpenid();

return recordRepository.findIsVoted(openid);

}

}

package com.miguo.matrix.entity.miniprogram;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import lombok.EqualsAndHashCode;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Table;

/\*\*

\* 功能描述：小程序轮播图

\*

\* @author Hocassian

\* @date 2019-11-28 09:15

\*/

@EqualsAndHashCode(callSuper = true)

@Data

@Slf4j

@Entity

@Accessors(chain = true)

@ApiModel("小程序轮播图实体")

@Table(name = "vote\_swipers")

public class MpSwiper extends BaseEntity {

@Column(name = "activity\_id")

@ApiModelProperty("活动id")

private String activityId;

@Column(name = "swiper\_name")

@ApiModelProperty("轮播图名称")

private String swiperName;

@Column(name = "swiper\_pic")

@ApiModelProperty("轮播图地址")

private String swiperPic;

}

package com.miguo.matrix.repository.miniprogram;

import com.miguo.matrix.entity.miniprogram.MpSwiper;

import com.miguo.matrix.vo.miniprogram.SwiperVo;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import java.util.List;

/\*\*

\* @author Hocassian

\*/

@Repository

public interface MpSwiperRepository extends JpaRepository<MpSwiper,String> {

/\*\*

\* 分页查找所有标题或者内容包含该关键字且未被软删除的活动（录入活动页面用,「keywords」为空时返回所有）

\* @param keywords

\* @param pageable

\* @return

\*/

@Query("SELECT new com.miguo.matrix.vo.miniprogram.SwiperVo(" +

"g.id," +

"g.createBy," +

"g.updateBy," +

"g.createAt," +

"g.updateAt," +

"g.isDel," +

"g.activityId," +

"g.swiperName," +

"g.swiperPic," +

"a.activityName) " +

"FROM MpSwiper g " +

"LEFT JOIN Activity a ON g.activityId=a.id WHERE g.swiperName LIKE %:#{#keywords}%")

Page<SwiperVo> findSwiperByKeywords(String keywords, Pageable pageable);

/\*\*

\* 通过id找轮播图

\* @param id

\* @return

\*/

MpSwiper findSwiperById(String id);

/\*\*

\* 通过id找激活的投票对象

\* @param id

\* @return

\*/

@Query(value = "select \* from vote\_swipers WHERE activity\_id = :#{#id}",nativeQuery = true)

List<MpSwiper> findActiveSwiper(String id);

}

package com.miguo.matrix.service.miniprogram;

import com.miguo.matrix.entity.miniprogram.MpSwiper;

import com.miguo.matrix.repository.miniprogram.MpSwiperRepository;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import com.miguo.matrix.vo.miniprogram.SwiperVo;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import javax.servlet.http.HttpSession;

import java.util.Date;

import java.util.List;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-12-17 10:02

\*/

@Service

public class MpSwiperService {

@Autowired

private MpSwiperRepository swiperRepository;

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

private HttpSession session;

public void add(MpSwiper mpSwiper){

mpSwiper.setCreateAt(new Date());

mpSwiper.setUpdateAt(new Date());

mpSwiper.setId(snowflakeIdWorker.nextId());

mpSwiper.setIsDel(false);

mpSwiper.setCreateBy((String) session.getAttribute("user"));

mpSwiper.setUpdateBy((String) session.getAttribute("user"));

swiperRepository.save(mpSwiper);

}

public void delete(List<MpSwiper> list){

swiperRepository.deleteInBatch(list);

}

public Page<SwiperVo> findSwiperByKeywords(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "updateAt");

return swiperRepository.findSwiperByKeywords(keywords,pageable);

}

public MpSwiper findOneById(String id)

{

return swiperRepository.findSwiperById(id);

}

public void update(MpSwiper mpSwiper){

MpSwiper mpSwiperTemp =this.findOneById(mpSwiper.getId());

BeanUtils.copyProperties(mpSwiper, mpSwiperTemp);

mpSwiperTemp.setUpdateBy((String) session.getAttribute("user"));

mpSwiperTemp.setUpdateAt(new Date());

swiperRepository.saveAndFlush(mpSwiperTemp);

}

public List<MpSwiper> findActiveOne(String id) {

return swiperRepository.findActiveSwiper(id);

}

}

package com.miguo.matrix.entity.miniprogram;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import lombok.EqualsAndHashCode;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Table;

import java.util.Date;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 10:25

\*/

@EqualsAndHashCode(callSuper = true)

@Slf4j

@Data

@Entity

@Accessors(chain = true)

@ApiModel("小程序公告栏实体")

@Table(name = "vote\_notes")

public class Note extends BaseEntity {

@Column(name = "activity\_id")

@ApiModelProperty("活动id")

private String activityId;

@Column(name = "note\_name")

@ApiModelProperty("公告项目名称")

private String noteName;

@Column(name = "note\_display\_time")

@ApiModelProperty("公告展示时间")

private Date noteDisplayTime;

@Column(name = "note\_details")

@ApiModelProperty("公告正文")

private String noteDetails;

}

package com.miguo.matrix.vo.miniprogram;

import com.fasterxml.jackson.annotation.JsonFormat;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import java.util.Date;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 10:25

\*/

@Data

@Slf4j

@NoArgsConstructor

@AllArgsConstructor

@Accessors(chain = true)

@ApiModel("小程序公告栏Vo实体")

public class NoteVo {

private String id;

private String createBy;

private String updateBy;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date createAt;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date updateAt;

private Boolean isDel = Boolean.FALSE;

@ApiModelProperty("活动id")

private String activityId;

@ApiModelProperty("公告项目名称")

private String noteName;

@ApiModelProperty("公告展示时间")

private Date noteDisplayTime;

@ApiModelProperty("公告正文")

private String noteDetails;

@ApiModelProperty("活动名称")

private String activityName;

}

package com.miguo.matrix.dto;

import lombok.Data;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2020-03-01 16:58

\*/

@Data

public class OpenIdJson {

private String openid;

private String session\_key;

private String unionid;

}

package com.miguo.matrix.dto;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import lombok.experimental.Accessors;

import java.io.Serializable;

import java.util.List;

/\*\*

\* @author Hocassian

\*/

@ApiModel("统一返回分页Dto")

@NoArgsConstructor

@AllArgsConstructor

@Data

@Accessors(chain = true)

public class PageResult<T> implements Serializable {

@ApiModelProperty("页码")

private Integer page;

@ApiModelProperty("页大小")

private Integer size;

@ApiModelProperty("总数")

private Long total;

@ApiModelProperty("数据")

private List<T> data;

}

package com.miguo.matrix.controller;

import com.miguo.matrix.dto.Result;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import net.coobird.thumbnailator.Thumbnails;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.\*;

import org.springframework.web.multipart.MultipartFile;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.IOException;

/\*\*

\* @author Hocassian

\*/

@Api("专门用来上传图片")

@RequestMapping("/picture")

@RestController

public class PictureController {

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Value("${file.path}")

private String filePath;

// 上传的路径

@ApiOperation("上传图片专用接口")

@PostMapping("/upload")

@CrossOrigin

public Result<String> upload(@RequestParam(value = "file") MultipartFile file){

Result<String> result = new Result<>();

// 文件名

String fileName = file.getOriginalFilename();

// 后缀名

assert fileName != null;

String suffixName = fileName.substring(fileName.lastIndexOf("."));

// 新文件名

fileName = snowflakeIdWorker.nextId() + suffixName;

File dest = new File(filePath + fileName);

if (!dest.getParentFile().exists()) {

// 若不存在该文件夹，则创建一个

boolean e = dest.getParentFile().mkdirs();

}

try {

file.transferTo(dest);

} catch (IOException e) {

e.printStackTrace();

}

System.out.println("返回的文件URL：" + fileName);

return result.setData(fileName).setMessage("上传成功！").setCode(HttpStatus.OK);

}

@ApiOperation("上传小程序投票对象图片专用接口")

@PostMapping("/upload\_mp")

@CrossOrigin

public Result<String> uploadMiniprogramPic(@RequestParam(value = "file") MultipartFile file){

Result<String> result = new Result<>();

// 文件名

String fileName = file.getOriginalFilename();

// 后缀名

assert fileName != null;

String suffixName = fileName.substring(fileName.lastIndexOf("."));

// 新文件名

fileName = snowflakeIdWorker.nextId() + suffixName;

File dest = new File(filePath + fileName);

if (!dest.getParentFile().exists()) {

// 若不存在该文件夹，则创建一个

boolean e = dest.getParentFile().mkdirs();

}

try {

file.transferTo(dest);

} catch (IOException e) {

e.printStackTrace();

}

File thumbnailDest = new File(filePath+ "thumbnails/mp" + fileName);

if (!thumbnailDest.getParentFile().exists()) {

// 若不存在该文件夹，则创建一个

boolean e = thumbnailDest.getParentFile().mkdirs();

}

try {

Thumbnails.of(dest)

.size(160,90)

.outputQuality(0.9f)

.toFile(thumbnailDest);

} catch (IOException e) {

e.printStackTrace();

}

System.out.println("返回的文件URL：" + fileName);

return result.setData(fileName).setMessage("上传成功！").setCode(HttpStatus.OK);

}

}

package com.miguo.matrix.controller;

import com.miguo.matrix.dto.Result;

import com.miguo.matrix.repository.staff.QualityRepository;

import com.miguo.matrix.service.staff.CountService;

import com.miguo.matrix.vo.CountVo;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import javax.servlet.http.HttpServletRequest;

import java.util.List;

import java.util.Map;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2020-03-25 14:28

\*/

@Api("统计专用接口")

@Slf4j

@RestController

@RequestMapping("/count")

public class QualityController {

@Autowired

private CountService countService;

@ApiOperation("统计所有数据")

@PostMapping("/quality")

public Result<Map<String, Integer>> findAllCount(String a) {

Result<Map<String, Integer>> result = new Result<>();

Map<String, Integer> map;

try {

map = countService.findAllCount();

result.setMessage("success").setCode(HttpStatus.OK).setData(map);

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("fail").setData(null);

}

return result;

}

}

package com.miguo.matrix.repository.staff;

import com.miguo.matrix.entity.client.Article;

import com.miguo.matrix.vo.CountVo;

import org.springframework.data.domain.Page;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import java.util.List;

import java.util.Map;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2020-03-25 01:03

\*/

@Repository

public interface QualityRepository extends JpaRepository<Article, String> {

// /\*\*

// \* 统计文章数量

// \* @return

// \*/

// @Query(value = "SELECT count(\*) from client\_article",nativeQuery = true)

// Integer countArticle();

//

// /\*\*

// \* 统计视频数量

// \* @return

// \*/

// @Query(value = "SELECT count(\*) from client\_video",nativeQuery = true)

// Integer countVideo();

//

// /\*\*

// \* 统计轮播图数量

// \* @return

// \*/

// @Query(value = "SELECT count(\*) from client\_swiper",nativeQuery = true)

// Integer countSwiper();

/\*\*

\* 统计所有数据

\* @return

\*/

@Query(value = "select t1.articleWebCount, t2.videoWebCount, t3.swiperWebCount, t4.viewWebCount, " +

"t5.activityMiniCount, t6.userMiniCount, t7.merchantMiniCount, t8.staffWebCount from " +

"(select count(\*) articleWebCount from client\_article) t1, " +

"(select count(\*) videoWebCount from client\_video) t2, " +

"(select count(\*) swiperWebCount from client\_swiper) t3, " +

"(select count(\*) viewWebCount from client\_swiper) t4, " +

"(select count(\*) activityMiniCount from vote\_activities) t5, " +

"(select count(\*) userMiniCount from vote\_records) t6, " +

"(select count(\*) merchantMiniCount from vote\_merchants) t7, " +

"(select count(\*) staffWebCount from com\_staff) t8" ,nativeQuery = true)

Map<String, Integer> countAll();

}

package com.miguo.matrix.config;

import org.springframework.amqp.core.Binding;

import org.springframework.amqp.core.BindingBuilder;

import org.springframework.amqp.core.DirectExchange;

import org.springframework.amqp.core.Queue;

import org.springframework.amqp.rabbit.core.RabbitTemplate;

import org.springframework.amqp.support.converter.Jackson2JsonMessageConverter;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.amqp.rabbit.connection.ConnectionFactory;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2020-03-07 13:26

\*/

@Configuration

public class RabbitConfig {

@Bean

public RabbitTemplate rabbitTemplate(final ConnectionFactory connectionFactory) {

final RabbitTemplate rabbitTemplate = new RabbitTemplate(connectionFactory);

rabbitTemplate.setMessageConverter(producerJackson2MessageConverter());

return rabbitTemplate;

}

@Bean

public Queue videoSendingQueue(){

return new Queue("videoSendingQueue",true);

}

DirectExchange videoSendingExchange(){

return new DirectExchange("videoSendingExchange");

}

@Bean

Binding bindingVideoSending(){

return BindingBuilder.bind(videoSendingQueue()).to(videoSendingExchange()).with("videoSendingRouting");

}

@Bean

public Jackson2JsonMessageConverter producerJackson2MessageConverter() {

return new Jackson2JsonMessageConverter();

}

}

package com.miguo.matrix.entity.miniprogram;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import lombok.EqualsAndHashCode;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Table;

import java.util.Date;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 10:35

\*/

@EqualsAndHashCode(callSuper = true)

@Slf4j

@Data

@Entity

@Accessors(chain = true)

@ApiModel("小程序投票记录实体")

@Table(name = "vote\_records")

public class Record extends BaseEntity {

@Column(name = "activity\_id")

@ApiModelProperty("活动id")

private String activityId;

@Column(name = "group\_id")

@ApiModelProperty("投票对象id")

private String groupId;

@Column(name = "record\_nickname")

@ApiModelProperty("用户昵称")

private String recordNickname;

@Column(name = "record\_openid")

@ApiModelProperty("用户openid")

private String recordOpenid;

@Column(name = "record\_date")

@ApiModelProperty("投票时间")

// @JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date recordDate;

}

package com.miguo.matrix.vo.miniprogram;

import com.fasterxml.jackson.annotation.JsonFormat;

import com.miguo.matrix.vo.BaseVo;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.EqualsAndHashCode;

import lombok.NoArgsConstructor;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import java.util.Date;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 10:35

\*/

@Slf4j

@Data

@NoArgsConstructor

@AllArgsConstructor

@Accessors(chain = true)

@ApiModel("小程序投票记录Vo实体")

public class RecordVo {

private String id;

private String createBy;

private String updateBy;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date createAt;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date updateAt;

private Boolean isDel = Boolean.FALSE;

@ApiModelProperty("活动id")

private String activityId;

@ApiModelProperty("投票对象id")

private String groupId;

@ApiModelProperty("用户昵称")

private String recordNickname;

@ApiModelProperty("用户openid")

private String recordOpenid;

@ApiModelProperty("投票时间")

// @JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date recordDate;

@ApiModelProperty("组名")

private String groupName;

}

package com.miguo.matrix.dto;

import com.fasterxml.jackson.databind.annotation.JsonSerialize;

import com.miguo.matrix.utils.EnumSerializer;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import lombok.experimental.Accessors;

import org.springframework.http.HttpStatus;

import java.io.Serializable;

/\*\*

\* @author Hocassian

\*/

@ApiModel("统一返回Dto")

@NoArgsConstructor

@AllArgsConstructor

@Data

@Accessors(chain = true)

public class Result<T> implements Serializable {

@ApiModelProperty("响应码")

@JsonSerialize(using = EnumSerializer.class)

private HttpStatus code;

@ApiModelProperty("描述")

private String message;

@ApiModelProperty("数据")

private T data;

}

package com.miguo.matrix.dto;

import lombok.Data;

import org.springframework.data.domain.Sort;

/\*\*

\* @author Hocassian

\*/

@Data

public class SearchDto {

private int page;

private int size;

private String keywords;

private Sort.Direction direction;

}

package com.miguo.matrix.utils;

import org.springframework.stereotype.Component;

/\*\*

\* 描述: Twitter的分布式自增ID雪花算法snowflake (Java版)

\* https://github.com/souyunku/SnowFlake

\* @create 2018-03-13 12:37

\*\*/

@Component

public class SnowflakeIdWorker {

/\*\*

\* 起始的时间戳

\*/

private final static long START\_STMP = 1480166465631L;

// private final static long START\_STMP = System.currentTimeMillis();

/\*\*

\* 每一部分占用的位数

\*/

private final static long SEQUENCE\_BIT = 12; //序列号占用的位数

private final static long MACHINE\_BIT = 5; //机器标识占用的位数

private final static long DATACENTER\_BIT = 5;//数据中心占用的位数

/\*\*

\* 每一部分的最大值

\*/

private final static long MAX\_DATACENTER\_NUM = -1L ^ (-1L << DATACENTER\_BIT);

private final static long MAX\_MACHINE\_NUM = -1L ^ (-1L << MACHINE\_BIT);

private final static long MAX\_SEQUENCE = -1L ^ (-1L << SEQUENCE\_BIT);

/\*\*

\* 每一部分向左的位移

\*/

private final static long MACHINE\_LEFT = SEQUENCE\_BIT;

private final static long DATACENTER\_LEFT = SEQUENCE\_BIT + MACHINE\_BIT;

private final static long TIMESTMP\_LEFT = DATACENTER\_LEFT + DATACENTER\_BIT;

private long datacenterId; //数据中心

private long machineId; //机器标识

private long sequence = 0L; //序列号

private long lastStmp = -1L;//上一次时间戳

public SnowflakeIdWorker() {

if (2 > MAX\_DATACENTER\_NUM || 2 < 0) {

throw new IllegalArgumentException("datacenterId can't be greater than MAX\_DATACENTER\_NUM or less than 0");

}

if (3 > MAX\_MACHINE\_NUM || 3 < 0) {

throw new IllegalArgumentException("machineId can't be greater than MAX\_MACHINE\_NUM or less than 0");

}

this.datacenterId = 2;

this.machineId = 3;

}

/\*\*

\* 产生下一个ID

\*我改返回类型，long->String

\* @return

\*/

public synchronized String nextId() {

long currStmp = getNewstmp();

if (currStmp < lastStmp) {

throw new RuntimeException("Clock moved backwards. Refusing to generate id");

}

if (currStmp == lastStmp) {

//相同毫秒内，序列号自增

sequence = (sequence + 1) & MAX\_SEQUENCE;

//同一毫秒的序列数已经达到最大

if (sequence == 0L) {

currStmp = getNextMill();

}

} else {

//不同毫秒内，序列号置为0

sequence = 0L;

}

lastStmp = currStmp;

return String.valueOf((currStmp - START\_STMP) << TIMESTMP\_LEFT //时间戳部分

| datacenterId << DATACENTER\_LEFT //数据中心部分

| machineId << MACHINE\_LEFT //机器标识部分

| sequence); //序列号部分

}

private long getNextMill() {

long mill = getNewstmp();

while (mill <= lastStmp) {

mill = getNewstmp();

}

return mill;

}

private long getNewstmp() {

return System.currentTimeMillis();

}

}

package com.miguo.matrix.controller;

import com.miguo.matrix.dto.PageResult;

import com.miguo.matrix.dto.Result;

import com.miguo.matrix.dto.SearchDto;

import com.miguo.matrix.entity.client.Article;

import com.miguo.matrix.entity.client.Swiper;

import com.miguo.matrix.entity.client.Video;

import com.miguo.matrix.entity.staff.Account;

import com.miguo.matrix.service.client.ArticleService;

import com.miguo.matrix.service.client.SwiperService;

import com.miguo.matrix.service.client.VideoService;

import com.miguo.matrix.service.staff.AccountService;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.\*;

import javax.servlet.http.HttpSession;

/\*\*

\* @author Hocassian

\*/

@Api("员工管理接口，对视频、文章进行增删改")

@Slf4j

@RestController

@RequestMapping("/staff")

public class StaffController {

@Autowired

private ArticleService articleService;

@Autowired

private VideoService videoService;

@Autowired

private SwiperService swiperService;

@Autowired

private AccountService accountService;

@Autowired

private HttpSession session;

// -----------以下为员工信息的增删改查----------

@ApiOperation("获取当前员工信息")

@PostMapping("/account/find\_one")

public Result<Account> findOneStaff(String a){

Result<Account> result = new Result<>();

try {

Account account = accountService.findOne((String) session.getAttribute("user"));

result.setMessage("success").setCode(HttpStatus.OK).setData(account);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation(value = "修改员工信息")

@PostMapping("/account/update")

public Result<String> updateStaffInfo(@RequestBody Account account) {

Result<String> result = new Result<>();

result.setMessage("update").setCode(HttpStatus.OK);

try {

accountService.update(account);

result.setData("success");

} catch (Exception e) {

result.setData("fail");

}

return result;

}

// -----------以下为文章的增删改查----------

@ApiOperation("文章的增加")

@PostMapping("/article/add")

public Result<String> articleAdd(@RequestBody Article article){

Result<String> result = new Result<>();

result.setMessage("add").setCode(HttpStatus.OK);

try{

String id = articleService.add(article);

result.setData(id).setCode(HttpStatus.OK).setMessage("success");

}catch (Exception e){

result.setData("fail");

}

return result;

}

@ApiOperation("批量下架文章")

@DeleteMapping("/article/delete/{ids}")

public Result<String> articleDelete(@PathVariable("ids") String ids){

Result<String> result = new Result<>();

result.setCode(HttpStatus.OK).setMessage("delete");

try{

articleService.delete(ids);

result.setData("success");

}catch (Exception e){

result.setData("fail");

}

return result;

}

@ApiOperation("删除单篇文章")

@DeleteMapping("/article/delete\_one/{id}")

public Result<String> articleDeleteOne(@PathVariable("id") String id){

Result<String> result = new Result<>();

result.setCode(HttpStatus.OK).setMessage("delete");

try{

articleService.deleteOne(id);

result.setData("success");

}catch (Exception e){

result.setData("fail");

}

return result;

}

@ApiOperation("更新文章")

@PutMapping("/article/update")

public Result<String> articleUpdate(@RequestBody Article article) {

Result<String> result = new Result<>();

result.setCode(HttpStatus.OK).setMessage("update");

try{

articleService.update(article);

result.setData("success");

}catch (Exception e){

result.setData("fail");

}

return result;

}

@ApiOperation("分页查找所有包含关键字的文章")

@PostMapping("/article/find\_all\_by\_keywords")

public Result<PageResult<Article>> articleFindAllByKeywords(@RequestBody SearchDto searchDto) {

Result<PageResult<Article>> result = new Result<>();

Page<Article> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = articleService.staffFindAllByKeywords("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

// 当关键字为空时，查询所有

} else {

page = articleService.staffFindAllByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<Article> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("获取某一篇文章")

@GetMapping("/article/find\_one\_by\_id/{id}")

public Result<Article> findArticleById(@PathVariable("id") String id){

Result<Article> result = new Result<>();

try {

Article list = articleService.findOneById(id);

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

// --------以下为视频的增删改查-------

@ApiOperation("视频的添加")

@PostMapping("/video/add")

public Result<String> videoAdd(@RequestBody Video video){

Result<String> result = new Result<>();

try{

videoService.add(video);

result.setCode(HttpStatus.OK).setMessage("add").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("add").setData("fail");

}

return result;

}

@ApiOperation("视频的批量下架")

@DeleteMapping("/video/delete/{ids}")

public Result<String> videoDelete(@PathVariable("ids") String ids) {

Result<String> result =new Result<>();

try{

videoService.delete(ids);

result.setCode(HttpStatus.OK).setMessage("delete").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("delete").setData("fail");

}

return result;

}

@ApiOperation("删除单个视频")

@DeleteMapping("/video/delete\_one/{id}")

public Result<String> videoDeleteOne(@PathVariable("id") String id){

Result<String> result = new Result<>();

result.setCode(HttpStatus.OK).setMessage("delete");

try{

videoService.deleteOne(id);

result.setData("success");

}catch (Exception e){

result.setData("fail");

}

return result;

}

@ApiOperation("视频的更新")

@PutMapping("/video/update")

public Result<String> videoUpdate(@RequestBody Video video){

Result<String> result = new Result<>();

try{

videoService.update(video);

result.setCode(HttpStatus.OK).setMessage("update").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("update").setData("fail");

}

return result;

}

@ApiOperation("分页查找所有模糊查询的视频")

@PostMapping("/video/find\_all\_by\_keywords")

public Result<PageResult<Video>> videoFindAllByKeywords(@RequestBody SearchDto searchDto) {

Result<PageResult<Video>> result = new Result<>();

Page<Video> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = videoService.staffFindAllByKeywords("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

// 当关键字为空时，查询所有

} else {

page = videoService.staffFindAllByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<Video> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("获取某一个视频")

@GetMapping("/video/find\_one\_by\_id/{id}")

public Result<Video> findVideoById(@PathVariable("id") String id){

Result<Video> result = new Result<>();

try {

Video list = videoService.findOneById(id);

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

// ----------以下为网站轮播图的增删改-------

@ApiOperation("网站轮播图的添加")

@PostMapping("/web\_swiper/add")

public Result<String> webSwiperAdd(@RequestBody Swiper swiper){

Result<String> result = new Result<>();

try{

swiperService.add(swiper);

result.setCode(HttpStatus.OK).setMessage("add").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("add").setData("fail");

}

return result;

}

@ApiOperation("网站轮播图的批量下架")

@DeleteMapping("/web\_swiper/delete/{ids}")

public Result<String> webSwiperDelete(@PathVariable("ids") String ids) {

Result<String> result =new Result<>();

try{

swiperService.deleteSome(ids);

result.setCode(HttpStatus.OK).setMessage("delete").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("delete").setData("fail");

}

return result;

}

@ApiOperation("网站轮播图的更新")

@PutMapping("/web\_swiper/update")

public Result<String> webSwiperUpdate(@RequestBody Swiper swiper){

Result<String> result = new Result<>();

try{

swiperService.update(swiper);

result.setCode(HttpStatus.OK).setMessage("update").setData("success");

}catch (Exception e){

System.out.print(e);

result.setCode(HttpStatus.OK).setMessage("update").setData("fail");

}

return result;

}

@ApiOperation("分页查找所有已被下架的网站轮播图")

@GetMapping("/web\_swiper/find\_all\_deleted/{page}/{size}")

public Result<PageResult<Swiper>> webSwiperFindAllDeleted(@PathVariable("page") int page, @PathVariable("size") int size){

Result<PageResult<Swiper>> result = new Result<>();

Page<Swiper> pageTemp;

try {

pageTemp = swiperService.findAllDeleted(page, size);

PageResult<Swiper> pageResult = new PageResult<>();

pageResult.setTotal(pageTemp.getTotalElements()).setData(pageTemp.getContent()).setPage(page).setSize(size);

result.setData(pageResult).setCode(HttpStatus.OK).setMessage("success");

} catch (Exception e) {

result.setData(null).setCode(HttpStatus.OK).setMessage("fail");

}

return result;

}

@ApiOperation("分页查找所有未被删除的网站轮播图")

@GetMapping("/web\_swiper/find\_all\_exist/{page}/{size}")

public Result<PageResult<Swiper>> webSwiperFindAllExist(@PathVariable("page") int page, @PathVariable("size") int size){

Result<PageResult<Swiper>> result = new Result<>();

Page<Swiper> pageTemp;

try {

pageTemp = swiperService.findAllExist(page, size);

PageResult<Swiper> pageResult = new PageResult<>();

pageResult.setTotal(pageTemp.getTotalElements()).setData(pageTemp.getContent()).setPage(page).setSize(size);

result.setData(pageResult).setCode(HttpStatus.OK).setMessage("success");

} catch (Exception e) {

result.setData(null).setCode(HttpStatus.OK).setMessage("fail");

}

return result;

}

@ApiOperation("分页查找所有模糊查询的视频")

@PostMapping("/web\_swiper/find\_all\_by\_keywords")

public Result<PageResult<Swiper>> webSwiperFindAllByKeywords(@RequestBody SearchDto searchDto) {

Result<PageResult<Swiper>> result = new Result<>();

Page<Swiper> page;

try {

if (searchDto.getKeywords() == null || searchDto.getKeywords().equals("")) {

page = swiperService.staffFindAllByKeywords("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

// 当关键字为空时，查询所有

} else {

page = swiperService.staffFindAllByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<Swiper> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("获取某一个轮播图")

@GetMapping("/web\_swiper/find\_one\_by\_id/{id}")

public Result<Swiper> findSwiperById(@PathVariable("id") String id){

Result<Swiper> result = new Result<>();

try {

Swiper list = swiperService.findOneById(id).get();

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

}

package com.miguo.matrix.controller.miniprogram;

import com.miguo.matrix.dto.PageResult;

import com.miguo.matrix.dto.Result;

import com.miguo.matrix.dto.SearchDto;

import com.miguo.matrix.entity.miniprogram.\*;

import com.miguo.matrix.service.miniprogram.\*;

import com.miguo.matrix.vo.miniprogram.\*;

import io.swagger.annotations.Api;

import io.swagger.annotations.ApiOperation;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

import java.util.Optional;

/\*\*

\* @author Hocassian

\*/

@Api("员工添加小程序相关数据的接口")

@Slf4j

@RestController

@RequestMapping("/staff\_miniprogram")

public class StaffMiniProgramController {

@Autowired

private MpSwiperService swiperService;

@Autowired

private MpActivityService activityService;

@Autowired

private MpGroupService groupService;

@Autowired

private MpRecordService recordService;

@Autowired

private MpMerchantService merchantService;

@Autowired

private MpNoteService noteService;

// ----------以下为小程序轮播图的增删查改-------

@ApiOperation("小程序轮播图的添加")

@PostMapping("/mp\_swiper/add")

public Result<String> mpSwiperAdd(@RequestBody MpSwiper mpSwiper){

Result<String> result = new Result<>();

try{

swiperService.add(mpSwiper);

result.setCode(HttpStatus.OK).setMessage("add").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("add").setData("fail");

}

return result;

}

@ApiOperation("小程序轮播图的批量软删除")

@PostMapping("/mp\_swiper/delete")

public Result<String> mpSwiperDelete(@RequestBody List<MpSwiper> list) {

Result<String> result =new Result<>();

try{

swiperService.delete(list);

result.setCode(HttpStatus.OK).setMessage("delete").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("delete").setData("fail");

}

return result;

}

@ApiOperation("小程序轮播图的更新")

@PutMapping("/mp\_swiper/update")

public Result<String> mpSwiperUpdate(@RequestBody MpSwiper mpSwiper){

Result<String> result = new Result<>();

try{

swiperService.update(mpSwiper);

result.setCode(HttpStatus.OK).setMessage("update").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("update").setData("fail");

}

return result;

}

@ApiOperation("分页查找所有标题或者内容包含该关键字的小程序轮播图（录入活动页面用,「keywords」为空时返回所有）")

@PostMapping("/mp\_swiper/find\_all\_by\_keywords")

public Result<PageResult<SwiperVo>> mpSwiperFindAll(@RequestBody SearchDto searchDto) {

Result<PageResult<SwiperVo>> result = new Result<>();

Page<SwiperVo> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = swiperService.findSwiperByKeywords("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

} else {

page = swiperService.findSwiperByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<SwiperVo> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

// ----------以下为小程序活动的增删查改-------

@ApiOperation("小程序活动的添加")

@PostMapping("/mp\_activity/add")

public Result<String> mpActivityAdd(@RequestBody Activity activity){

Result<String> result = new Result<>();

try{

activityService.add(activity);

result.setCode(HttpStatus.OK).setMessage("add").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("add").setData("fail");

}

return result;

}

@ApiOperation("小程序活动的批量删除")

@PostMapping("/mp\_activity/delete")

public Result<String> mpActivityDelete(@RequestBody List<Activity> list) {

Result<String> result =new Result<>();

try{

activityService.delete(list);

result.setCode(HttpStatus.OK).setMessage("delete").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("delete").setData("fail");

}

return result;

}

@ApiOperation("小程序活动的更新")

@PutMapping("/mp\_activity/update")

public Result<String> mpActivityUpdate(@RequestBody Activity activity){

Result<String> result = new Result<>();

try{

activityService.update(activity);

result.setCode(HttpStatus.OK).setMessage("update").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("update").setData("fail");

}

return result;

}

@ApiOperation("分页查找所有标题或者内容包含该关键字的活动（录入活动页面用,「keywords」为空时返回所有）")

@PostMapping("/mp\_activity/find\_all\_by\_keywords")

public Result<PageResult<Activity>> mpActivityFindAll(@RequestBody SearchDto searchDto) {

Result<PageResult<Activity>> result = new Result<>();

Page<Activity> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = activityService.findAllActivityByKeywords("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

} else {

page = activityService.findAllActivityByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<Activity> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("不分页查找所有标题或者内容包含该关键字的活动")

@GetMapping("/mp\_activity/find\_all\_by\_keywords\_from\_input/{keyword}")

public Result<List<Activity>> mpActivityFindAllFromInput(@PathVariable("keyword") String keywords) {

Result<List<Activity>> result = new Result<>();

try {

List<Activity> list;

if (keywords == null || "".equals(keywords)) {

list = activityService.findAllActivityByKeywordsFromInput("");

} else {

list = activityService.findAllActivityByKeywordsFromInput(keywords);

}

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("用于返回该表联结的值")

@GetMapping("/mp\_activity/find\_activity\_id/{id}")

public Result<Optional<Activity>> findActivityById(@PathVariable("id") String id){

Optional<Activity> list= activityService.findById(id);

Result<Optional<Activity>> result =new Result<>();

result.setCode(HttpStatus.OK).setData(list).setMessage("success");

return result;

}

// ----------以下为小程序投票对象的增删查改-------

@ApiOperation("小程序投票对象的添加")

@PostMapping("/mp\_group/add")

public Result<String> mpGroupAdd(@RequestBody GroupVo groupVo){

Result<String> result = new Result<>();

try{

groupService.add(groupVo);

result.setCode(HttpStatus.OK).setMessage("add").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("add").setData("fail");

}

return result;

}

@ApiOperation("小程序投票对象的批量软删除")

@PostMapping("/mp\_group/delete")

public Result<String> mpGroupDelete(@RequestBody List<Group> list) {

Result<String> result =new Result<>();

try{

groupService.delete(list);

result.setCode(HttpStatus.OK).setMessage("delete").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("delete").setData("fail");

}

return result;

}

@ApiOperation("小程序投票对象的更新")

@PutMapping("/mp\_group/update")

public Result<String> mpGroupUpdate(@RequestBody Group group){

Result<String> result = new Result<>();

try{

groupService.update(group);

result.setCode(HttpStatus.OK).setMessage("update").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("update").setData("fail");

}

return result;

}

@ApiOperation("分页查找所有标题或者内容包含该关键字的小程序投票对象（录入活动页面用,「keywords」为空时返回所有）")

@PostMapping("/mp\_group/find\_all\_by\_keywords")

public Result<PageResult<GroupVo>> mpGroupFindAll(@RequestBody SearchDto searchDto) {

Result<PageResult<GroupVo>> result = new Result<>();

Page<GroupVo> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = groupService.findGroupByKeywords("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

} else {

page = groupService.findGroupByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<GroupVo> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

System.out.print(e);

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("不分页查找所有标题或者内容包含该关键字的小程序投票对象")

@GetMapping("/mp\_group/find\_all\_by\_keywords\_from\_input/{keyword}")

public Result<List<Group>> mpGroupFindAllFromInput(@PathVariable("keyword") String keywords) {

Result<List<Group>> result = new Result<>();

try {

List<Group> list;

if (keywords == null || "".equals(keywords)) {

list = groupService.findAllActivityByKeywordsFromInput("");

} else {

list = groupService.findAllActivityByKeywordsFromInput(keywords);

}

result.setMessage("success").setCode(HttpStatus.OK).setData(list);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

@ApiOperation("用于返回该表联结的值")

@GetMapping("/mp\_group/find\_group\_id/{id}")

public Result<Optional<Group>> findGroupById(@PathVariable("id") String id){

Optional<Group> list= groupService.findById(id);

Result<Optional<Group>> result =new Result<>();

result.setCode(HttpStatus.OK).setData(list).setMessage("success");

return result;

}

// ----------以下为小程序投票记录的增删查改-------

@ApiOperation("小程序投票记录的添加")

@PostMapping("/mp\_record/add")

public Result<String> mpRecordAdd(@RequestBody Record record){

Result<String> result = new Result<>();

try{

recordService.add(record);

result.setCode(HttpStatus.OK).setMessage("add").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("add").setData("fail");

}

return result;

}

@ApiOperation("小程序投票记录的批量软删除")

@PostMapping("/mp\_record/delete")

public Result<String> mpRecordDelete(@RequestBody List<Record> list) {

Result<String> result =new Result<>();

try{

recordService.delete(list);

result.setCode(HttpStatus.OK).setMessage("delete").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("delete").setData("fail");

}

return result;

}

@ApiOperation("小程序投票记录的更新")

@PutMapping("/mp\_record/update")

public Result<String> mpRecordUpdate(@RequestBody Record record){

Result<String> result = new Result<>();

try{

recordService.update(record);

result.setCode(HttpStatus.OK).setMessage("update").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("update").setData("fail");

}

return result;

}

@ApiOperation("分页查找所有标题或者内容包含该关键字的小程序投票记录（录入活动页面用,「keywords」为空时返回所有）")

@PostMapping("/mp\_record/find\_all\_by\_keywords")

public Result<PageResult<RecordVo>> mpRecordFindAll(@RequestBody SearchDto searchDto) {

Result<PageResult<RecordVo>> result = new Result<>();

Page<RecordVo> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = recordService.findRecordByKeywords("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

} else {

page = recordService.findRecordByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<RecordVo> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

// ----------以下为小程序赞助商的增删查改-------

@ApiOperation("小程序赞助商的添加")

@PostMapping("/mp\_merchant/add")

public Result<String> mpMerchantAdd(@RequestBody Merchant merchant){

Result<String> result = new Result<>();

try{

merchantService.add(merchant);

result.setCode(HttpStatus.OK).setMessage("add").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("add").setData("fail");

}

return result;

}

@ApiOperation("小程序赞助商的批量删除")

@PostMapping("/mp\_merchant/delete")

public Result<String> mpMerchantDelete(@RequestBody List<Merchant> list) {

Result<String> result =new Result<>();

try{

merchantService.delete(list);

result.setCode(HttpStatus.OK).setMessage("delete").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("delete").setData("fail");

}

return result;

}

@ApiOperation("小程序赞助商的更新")

@PutMapping("/mp\_merchant/update")

public Result<String> mpMerchantUpdate(@RequestBody Merchant merchant){

Result<String> result = new Result<>();

try{

merchantService.update(merchant);

result.setCode(HttpStatus.OK).setMessage("update").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("update").setData("fail");

}

return result;

}

@ApiOperation("分页查找所有标题或者内容包含该关键字的小程序赞助商（录入活动页面用,「keywords」为空时返回所有）")

@PostMapping("/mp\_merchant/find\_all\_by\_keywords")

public Result<PageResult<MerchantVo>> mpMerchantFindAll(@RequestBody SearchDto searchDto) {

Result<PageResult<MerchantVo>> result = new Result<>();

Page<MerchantVo> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = merchantService.findMerchantByKeywords("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

} else {

page = merchantService.findMerchantByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<MerchantVo> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

System.out.print(e);

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

// ----------以下为小程序公告栏的增删查改-------

@ApiOperation("小程序公告栏的添加")

@PostMapping("/mp\_note/add")

public Result<String> mpNoteAdd(@RequestBody Note note){

Result<String> result = new Result<>();

try{

noteService.add(note);

result.setCode(HttpStatus.OK).setMessage("add").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("add").setData("fail");

}

return result;

}

@ApiOperation("小程序公告栏的批量软删除")

@PostMapping("/mp\_note/delete")

public Result<String> mpNoteDelete(@RequestBody List<Note> list) {

Result<String> result =new Result<>();

try{

noteService.delete(list);

result.setCode(HttpStatus.OK).setMessage("delete").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("delete").setData("fail");

}

return result;

}

@ApiOperation("小程序公告栏的更新")

@PutMapping("/mp\_note/update")

public Result<String> mpNoteUpdate(@RequestBody Note note){

Result<String> result = new Result<>();

try{

noteService.update(note);

result.setCode(HttpStatus.OK).setMessage("update").setData("success");

}catch (Exception e){

result.setCode(HttpStatus.OK).setMessage("update").setData("fail");

}

return result;

}

@ApiOperation("分页查找所有标题或者内容包含该关键字的小程序公告栏（录入活动页面用,「keywords」为空时返回所有）")

@PostMapping("/mp\_note/find\_all\_by\_keywords")

public Result<PageResult<NoteVo>> mpNoteFindAll(@RequestBody SearchDto searchDto) {

Result<PageResult<NoteVo>> result = new Result<>();

Page<NoteVo> page;

try {

if (searchDto.getKeywords() == null || "".equals(searchDto.getKeywords())) {

page = noteService.findNoteByKeywords("", searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

} else {

page = noteService.findNoteByKeywords(searchDto.getKeywords(), searchDto.getPage(), searchDto.getSize(), searchDto.getDirection());

}

PageResult<NoteVo> pageResult = new PageResult<>();

pageResult.setSize(searchDto.getSize()).setPage(searchDto.getPage()).setData(page.getContent()).setTotal(page.getTotalElements());

result.setMessage("success").setCode(HttpStatus.OK).setData(pageResult);

} catch (Exception e) {

result.setMessage("fail").setCode(HttpStatus.OK).setData(null);

}

return result;

}

}

package com.miguo.matrix.config;

import io.swagger.annotations.Api;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.SpringBootConfiguration;

import org.springframework.context.annotation.Bean;

import org.springframework.core.env.Environment;

import org.springframework.core.env.Profiles;

import springfox.documentation.builders.ApiInfoBuilder;

import springfox.documentation.builders.PathSelectors;

import springfox.documentation.builders.RequestHandlerSelectors;

import springfox.documentation.service.ApiInfo;

import springfox.documentation.service.Contact;

import springfox.documentation.spi.DocumentationType;

import springfox.documentation.spring.web.plugins.Docket;

import springfox.documentation.swagger2.annotations.EnableSwagger2;

/\*\*

\* 功能描述：SwaggerUI配置类

\* @author Hocassian

\* @Date 2019/11/26 10:21

\*/

@EnableSwagger2

@SpringBootConfiguration

public class SwaggerConfig {

/\*\*

\* 是否启动文档标识

\*/

private Boolean enableSwagger = Boolean.FALSE;

private Environment environment;

/\*\*

\* 功能描述：注入环境变量

\* @author Hocassian

\* @Date 2019/11/26 10:21

\*/

@Autowired

public void setEnvironment(Environment environment) {

this.environment = environment;

/\* 通过启动的环境变量决定是否启用Swagger-UI文档 \*/

Profiles profiles = Profiles.of("dev", "main");

enableSwagger = environment.acceptsProfiles(profiles);

}

@Bean("staffApi")

public Docket accountApi() {

return new Docket(DocumentationType.SWAGGER\_2)

.groupName("员工接口（官网）")

.select()

.apis(RequestHandlerSelectors.withClassAnnotation(Api.class))

.paths(PathSelectors.regex("/staff.\*"))

.build()

.apiInfo(apiInfo());

}

@Bean("mediaApi")

public Docket mediaApi() {

return new Docket(DocumentationType.SWAGGER\_2)

.groupName("员工接口（媒体分发）")

.select()

.apis(RequestHandlerSelectors.withClassAnnotation(Api.class))

.paths(PathSelectors.regex("/media.\*"))

.build()

.apiInfo(apiInfo());

}

@Bean("adminApi")

public Docket adminApi() {

return new Docket(DocumentationType.SWAGGER\_2)

.groupName("管理员接口（官网）")

.select()

.apis(RequestHandlerSelectors.withClassAnnotation(Api.class))

.paths(PathSelectors.regex("/admin.\*"))

.build()

.apiInfo(apiInfo());

}

@Bean("clientApi")

public Docket clientApi() {

return new Docket(DocumentationType.SWAGGER\_2)

.groupName("客户接口（官网）")

.select()

.apis(RequestHandlerSelectors.withClassAnnotation(Api.class))

.paths(PathSelectors.regex("/client.\*"))

.build()

.apiInfo(apiInfo());

}

@Bean("mpClientApi")

public Docket mpClientApi() {

return new Docket(DocumentationType.SWAGGER\_2)

.groupName("客户接口（小程序）")

.select()

.apis(RequestHandlerSelectors.withClassAnnotation(Api.class))

.paths(PathSelectors.regex("/client\_miniprogram.\*"))

.build()

.apiInfo(apiInfo());

}

@Bean("mpStaffApi")

public Docket mpStaffApi() {

return new Docket(DocumentationType.SWAGGER\_2)

.groupName("员工接口（小程序）")

.select()

.apis(RequestHandlerSelectors.withClassAnnotation(Api.class))

.paths(PathSelectors.regex("/staff\_miniprogram.\*"))

.build()

.apiInfo(apiInfo());

}

@Bean("loginApi")

public Docket loginApi() {

return new Docket(DocumentationType.SWAGGER\_2)

.groupName("登陆接口（官网）")

.select()

.apis(RequestHandlerSelectors.withClassAnnotation(Api.class))

.paths(PathSelectors.regex("/account.\*"))

.build()

.apiInfo(apiInfo());

}

@Bean("uploadApi")

public Docket uploadApi() {

return new Docket(DocumentationType.SWAGGER\_2)

.groupName("图片上传接口（综合）")

.select()

.apis(RequestHandlerSelectors.withClassAnnotation(Api.class))

.paths(PathSelectors.regex("/picture.\*"))

.build()

.apiInfo(apiInfo());

}

@Bean("countApi")

public Docket countApi() {

return new Docket(DocumentationType.SWAGGER\_2)

.groupName("數據統計")

.select()

.apis(RequestHandlerSelectors.withClassAnnotation(Api.class))

.paths(PathSelectors.regex("/count.\*"))

.build()

.apiInfo(apiInfo());

}

/\*\*

\* 功能描述：SwaggerUI文档的基本信息

\* @author Hocassian

\* @Date 2019/11/26 10:21

\*/

private ApiInfo apiInfo() {

return new ApiInfoBuilder()

.title("数据一览")

.description("所有的接口都在这里了~")

.termsOfServiceUrl("http://localhost:9090")

.contact(new Contact("Hocassian", "https://Hocassian.cn", "hokaso@qq.com"))

.version("v1.1.2")

.build();

}

}

package com.miguo.matrix.entity.client;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.EqualsAndHashCode;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import lombok.Data;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Table;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 11:50

\*/

@EqualsAndHashCode(callSuper = true)

@Data

@Slf4j

@Entity

@Accessors(chain = true)

@ApiModel("客户端轮播图实体")

@Table(name = "client\_swiper")

public class Swiper extends BaseEntity {

@Column(name = "swiper\_pic")

@ApiModelProperty("轮播图地址")

private String swiperPic;

@Column(name = "swiper\_name")

@ApiModelProperty("轮播图名称")

private String swiperName;

@Column(name = "swiper\_status")

@ApiModelProperty("轮播图名称")

private String swiperStatus;

@Column(name = "swiper\_reviewer")

@ApiModelProperty("轮播图审核人")

private String swiperReviewer;

}

package com.miguo.matrix.repository.client;

import com.miguo.matrix.entity.client.Swiper;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Modifying;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import java.util.Date;

import java.util.List;

/\*\*

\* @author Hocassian

\*/

@Repository

public interface SwiperRepository extends JpaRepository<Swiper,String> {

/\*\*

\* 通过把「is\_del」改为「true」来下架多个轮播图（并非删除）

\* @param id

\* @param date

\* @param updateBy

\*/

@Modifying

@Transactional(rollbackFor = Exception.class)

@Query(value = "update client\_swiper set is\_del = true , update\_at = :#{#date},update\_by = :#{#updateBy} where id = :#{#id} ", nativeQuery = true)

void deleteById(String id, Date date, String updateBy);

/\*\*

\* 分页返回标题或内容包含某关键字的文章条目（员工使用），当关键字为空时按更新时间顺序返回所有

\* @param keywords

\* @param pageable

\* @return

\*/

@Query(value = "select \* from client\_swiper WHERE swiper\_name LIKE %:#{#keywords}%",nativeQuery = true)

Page<Swiper> staffFindSwiperByKeywords(String keywords,Pageable pageable);

/\*\*

\* 查找所有已被软删除的轮播图

\* @param pageable

\* @return

\*/

@Query(value = "select \* from client\_swiper where is\_del = true",nativeQuery = true)

Page<Swiper> findAllDeletedSwiper(Pageable pageable);

/\*\*

\* 查找所有未被软删除的轮播图

\* @param pageable

\* @return

\*/

@Query(value = "select \* from client\_swiper where is\_del = false",nativeQuery = true)

Page<Swiper> findAllExistSwiper(Pageable pageable);

/\*\*

\* 查找所有未被软删除的轮播图（客户使用）

\* @param

\* @return

\*/

@Query(value = "select \* from client\_swiper where is\_del = false and swiper\_status = 'reviewed'",nativeQuery = true)

List<Swiper> clientFindAllExistSwiper();

/\*\*

\* 通过id找轮播图

\* @param id

\* @return

\*/

Swiper findWebSwiperById(String id);

/\*\*

\* ※审核方法

\*

\* 分页按条件分类查找删除与否的文章

\* @param pageable

\* @param active

\* @param keywords

\* @return

\*/

@Query(value = "select \* from client\_swiper where ( swiper\_name LIKE %:#{#keywords}% ) and is\_del = :active and swiper\_status != 'draft'", nativeQuery = true)

Page<Swiper> findAllClassSwiper(String keywords, Pageable pageable, Boolean active);

/\*\*

\* ※审核方法

\*

\* 分页返回所有的文章条目

\* @param pageable

\* @param keywords

\* @return

\*/

@Query(value = "select \* from client\_swiper where ( swiper\_name LIKE %:#{#keywords}% ) and swiper\_status != 'draft'", nativeQuery = true)

Page<Swiper> findAllExistSwiper(String keywords, Pageable pageable);

}

package com.miguo.matrix.service.client;

import com.miguo.matrix.entity.client.Swiper;

import com.miguo.matrix.repository.client.SwiperRepository;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import javax.servlet.http.HttpSession;

import java.util.Date;

import java.util.List;

import java.util.Optional;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-12-17 10:02

\*/

@Service

public class SwiperService {

@Autowired

private SwiperRepository swiperRepository;

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

private HttpSession session;

/\*\*

\* 添加文章

\* @param swiper

\*/

public void add(Swiper swiper){

swiper.setCreateAt(new Date());

swiper.setUpdateAt(new Date());

swiper.setId(snowflakeIdWorker.nextId());

swiper.setIsDel(false);

swiper.setCreateBy((String) session.getAttribute("user"));

swiper.setUpdateBy((String) session.getAttribute("user"));

swiperRepository.save(swiper);

}

/\*\*

\* 批量下架轮播图

\* @param ids

\*/

public void deleteSome(String ids){

String deleteIds = ids.split(",");

for (String deleteId : deleteIds) {

swiperRepository.deleteById(deleteId, new Date(), (String) session.getAttribute("user"));

}

}

/\*\*

\* 删除一个轮播图

\* @param id

\*/

public void deleteOne(String id){

swiperRepository.deleteById(id);

}

/\*\*

\* 分页返回标题或内容包含某关键字的文章条目（员工使用），当关键字为空时按更新时间顺序返回所有

\* @param keywords

\* @param page

\* @param size

\* @param direction

\* @return

\*/

public Page<Swiper> staffFindAllByKeywords(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "update\_at");

return swiperRepository.staffFindSwiperByKeywords(keywords,pageable);

}

/\*\*

\* 分页返回所有已被下架的轮播图条目

\* @param page

\* @param size

\* @return

\*/

public Page<Swiper> findAllDeleted(int page,int size){

page--;

Pageable pageable=PageRequest.of(page,size);

return swiperRepository.findAllDeletedSwiper(pageable);

}

/\*\*

\* 分页返回所有未被下架的轮播图条目

\* @param page

\* @param size

\* @return

\*/

public Page<Swiper> findAllExist(int page,int size){

page--;

Pageable pageable=PageRequest.of(page,size);

return swiperRepository.findAllExistSwiper(pageable);

}

/\*\*

\* 分页返回所有未被下架的轮播图条目

\* @param

\* @return

\*/

public List<Swiper> clientFindAllExist(){

return swiperRepository.clientFindAllExistSwiper();

}

/\*\*

\* 通过id找轮播图

\* @param id

\* @return

\*/

public Optional<Swiper> findOneById(String id)

{

return swiperRepository.findById(id);

}

/\*\*

\* 更新轮播图

\* @param swiper

\*/

public void update(Swiper swiper){

Swiper swiperTemp=this.findOneById(swiper.getId()).get();

BeanUtils.copyProperties(swiper, swiperTemp);

swiperTemp.setUpdateBy((String) session.getAttribute("user"));

swiperTemp.setUpdateAt(new Date());

swiperRepository.saveAndFlush(swiperTemp);

}

/\*\*

\* ※审核方法

\*

\* 分页分类返回所有条目

\* @param page

\* @param size

\* @return

\*/

public Page<Swiper> findAllClass(String keywords, int page,int size, String active, Sort.Direction direction){

page--;

Pageable pageable=PageRequest.of(page,size, direction, "create\_at");

Boolean activeTemp = Boolean.parseBoolean(active);

return swiperRepository.findAllClassSwiper(keywords, pageable, activeTemp);

}

/\*\*

\* ※审核方法

\*

\* 分页不分类返回所有条目

\* @param page

\* @param size

\* @return

\*/

public Page<Swiper> findAllExist(String keywords, int page,int size, Sort.Direction direction){

page--;

Pageable pageable=PageRequest.of(page,size, direction, "create\_at");

return swiperRepository.findAllExistSwiper(keywords, pageable);

}

}

package com.miguo.matrix.vo.miniprogram;

import com.fasterxml.jackson.annotation.JsonFormat;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import java.util.Date;

/\*\*

\* 功能描述：小程序轮播图

\*

\* @author Hocassian

\* @date 2019-11-28 09:15

\*/

@Data

@Slf4j

@NoArgsConstructor

@AllArgsConstructor

@Accessors(chain = true)

@ApiModel("小程序轮播图Vo实体")

public class SwiperVo{

private String id;

private String createBy;

private String updateBy;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date createAt;

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date updateAt;

private Boolean isDel = Boolean.FALSE;

@ApiModelProperty("活动id")

private String activityId;

@ApiModelProperty("轮播图名称")

private String swiperName;

@ApiModelProperty("轮播图地址")

private String swiperPic;

@ApiModelProperty("活动名称")

private String activityName;

}

package com.tocean.k12.test;

import com.miguo.matrix.Application;

import com.miguo.matrix.entity.client.Article;

import com.miguo.matrix.entity.client.Video;

import com.miguo.matrix.repository.client.ArticleRepository;

import com.miguo.matrix.repository.staff.AccountRepository;

import com.miguo.matrix.service.client.ArticleService;

import com.miguo.matrix.service.client.VideoService;

import com.miguo.matrix.service.staff.AccountService;

import org.junit.Test;

import org.junit.runner.RunWith;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.boot.test.context.SpringBootTest;

import org.springframework.test.context.junit4.SpringRunner;

import java.io.IOException;

@RunWith(SpringRunner.class)

@SpringBootTest(classes = {Application.class})

public class TestApplicationTest {

@Autowired

private AccountRepository accountRepository;

@Autowired

private AccountService accountService;

@Autowired

private ArticleService articleService;

@Autowired

private ArticleRepository ArticleRepository;

@Autowired

private VideoService videoService;

@Value("${file.path}")

private String filePath;

@Test

public void test() throws IOException {

System.out.println(System.getProperty("os.name"));

}

}

package com.miguo.matrix.dto.staff;

import lombok.Data;

@Data

public class UpdatePasswordDto {

// 原密码

private String beforePassword;

// 新密码

private String afterPassword;

}

package com.miguo.matrix.interceptor;

import com.fasterxml.jackson.databind.ObjectMapper;

import com.miguo.matrix.dto.Result;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.stereotype.Component;

import org.springframework.web.bind.annotation.RestController;

import org.springframework.web.servlet.HandlerInterceptor;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import java.io.IOException;

import java.io.OutputStream;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2020-03-08 23:33

\*/

@Component

public class UserInterceptor implements HandlerInterceptor {

@Autowired

private HttpSession session;

private static final String STAFF = "user";

@Override

public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handle) throws Exception {

if(session.getAttribute("user") !=null){

return true;

}

else{

response.setStatus(HttpStatus.UNAUTHORIZED.value());

OutputStream outputStream = response.getOutputStream();

outputStream.write((new ObjectMapper().writeValueAsString(new Result<String>().setCode(HttpStatus.UNAUTHORIZED).setMessage("noLogin").setData("noLogin"))).getBytes());

outputStream.flush();

outputStream.close();

return false;

}

}

}

package com.miguo.matrix.entity.client;

import com.fasterxml.jackson.annotation.JsonFormat;

import com.miguo.matrix.entity.BaseEntity;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import lombok.EqualsAndHashCode;

import lombok.experimental.Accessors;

import lombok.extern.slf4j.Slf4j;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Table;

import java.util.Date;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-28 11:53

\*/

@EqualsAndHashCode(callSuper = true)

@Slf4j

@Data

@Entity

@Accessors(chain = true)

@ApiModel("视频实体")

@Table(name = "client\_video")

public class Video extends BaseEntity {

@Column(name = "video\_url")

@ApiModelProperty("视频链接")

private String videoUrl;

@Column(name = "video\_cid")

@ApiModelProperty("视频cid")

private String videoCid;

@Column(name = "video\_title")

@ApiModelProperty("视频标题")

private String videoTitle;

@Column(name = "video\_profile")

@ApiModelProperty("视频简介")

private String videoProfile;

@Column(name = "video\_cover")

@ApiModelProperty("视频封面（横屏封面）")

private String videoCover;

@Column(name = "video\_pic")

@ApiModelProperty("视频封面")

private String videoPic;

@Column(name = "video\_date")

@ApiModelProperty("创作日期")

@JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone = "GMT+8")

private Date videoDate;

@Column(name = "video\_status")

@ApiModelProperty("状态")

private String videoStatus;

@Column(name = "video\_author")

@ApiModelProperty("创作团队")

private String videoAuthor;

@Column(name = "video\_reviewer")

@ApiModelProperty("审核人")

private String videoReviewer;

}

package com.miguo.matrix.repository.client;

import com.miguo.matrix.entity.client.Video;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Modifying;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import java.util.Date;

/\*\*

\* @author Hocassian

\*/

@Repository

public interface VideoRepository extends JpaRepository<Video,String> {

/\*\*

\* ※客户方法

\*

\* 分页返回视频标题或内容包含某关键字且未被下架的视频条目（客户使用）

\* @param keywords

\* @param pageable

\* @return

\*/

@Query(value = "select \* from client\_video WHERE ( video\_title LIKE %:#{#keywords}% OR video\_profile LIKE %:#{#keywords}% ) and is\_del = false and video\_status = 'reviewed'", nativeQuery = true)

Page<Video> findVideoByKeywords(String keywords, Pageable pageable);

/\*\*

\* ※员工方法：进入某一篇视频修改时使用

\*

\* 通过id找视频

\* @param id

\* @return

\*/

Video findVideoById(String id);

/\*\*

\* ※员工方法

\*

\* 分页返回视频标题或内容包含某关键字的视频条目，当关键字为空时按更新时间顺序返回所有

\* @param keywords

\* @param pageable

\* @return

\*/

@Query(value = "select \* from client\_video WHERE video\_title LIKE %:#{#keywords}% OR video\_profile LIKE %:#{#keywords}%", nativeQuery = true)

Page<Video> staffFindVideoByKeywords(String keywords, Pageable pageable);

/\*\*

\* ※审核方法：该方法与网络安全有关，与业务需求无关，未来迭代版本中使用

\*

\* 通过把「is\_del」改为「true」来下架多个视频（并非删除）

\* @param id

\* @param date

\* @param updateBy

\*/

@Modifying

@Transactional(rollbackFor = Exception.class)

@Query(value = "update client\_video set is\_del = true , update\_at = :#{#date},update\_by = :#{#updateBy} where id = :#{#id} ", nativeQuery = true)

void deleteSomeById(String id, Date date, String updateBy);

/\*\*

\* ※审核方法

\*

\* 分页不分类返回所有的视频条目

\* @param pageable

\* @param keywords

\* @return

\*/

@Query(value = "select \* from client\_video where ( video\_title LIKE %:#{#keywords}% OR video\_profile LIKE %:#{#keywords}% ) and video\_status != 'draft'", nativeQuery = true)

Page<Video> findAllExistVideo(String keywords, Pageable pageable);

/\*\*

\* ※审核方法

\*

\* 分页分类返回所有的视频条目

\* @param pageable

\* @param active

\* @param keywords

\* @return

\*/

@Query(value = "select \* from client\_video where ( video\_title LIKE %:#{#keywords}% OR video\_profile LIKE %:#{#keywords}% ) and is\_del = :active and video\_status != 'draft'",nativeQuery = true)

Page<Video> findAllClassVideo(String keywords, Pageable pageable, Boolean active);

}

package com.miguo.matrix.service.client;

import com.miguo.matrix.entity.client.Video;

import com.miguo.matrix.repository.client.VideoRepository;

import com.miguo.matrix.utils.SnowflakeIdWorker;

import org.springframework.beans.BeanUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Sort;

import org.springframework.stereotype.Service;

import javax.servlet.http.HttpSession;

import java.util.Date;

/\*\*

\* @author Hocassian

\*/

@Service

public class VideoService {

@Autowired

private SnowflakeIdWorker snowflakeIdWorker;

@Autowired

private VideoRepository videoRepository;

@Autowired

private HttpSession session;

/\*\*

\* 批量下架视频

\* @param ids

\*/

public void delete(String ids){

String deleteIds = ids.split(",");

for (String deleteId : deleteIds) {

videoRepository.deleteSomeById(deleteId, new Date(), (String) session.getAttribute("user"));

}

}

/\*\*

\* 添加视频

\* @param video

\*/

public void add(Video video){

video.setVideoDate(new Date());

video.setCreateAt(new Date());

video.setUpdateAt(new Date());

video.setUpdateBy((String) session.getAttribute("user"));

video.setCreateBy((String) session.getAttribute("user"));

video.setId(snowflakeIdWorker.nextId());

video.setIsDel(false);

videoRepository.save(video);

}

/\*\*

\* 分页返回标题或内容包含某关键字且未被下架的视频条目（客户使用）

\* @param keywords

\* @param page

\* @param size

\* @return

\*/

public Page<Video> findAllByKeywords(String keywords,int page,int size){

page--;

Pageable pageable = PageRequest.of(page, size);

return videoRepository.findVideoByKeywords(keywords,pageable);

}

/\*\*

\* 分页返回标题或内容包含某关键字的视频条目（员工使用），当关键字为空时按更新时间顺序返回所有

\* @param keywords

\* @param page

\* @param size

\* @param direction

\* @return

\*/

public Page<Video> staffFindAllByKeywords(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "update\_at");

return videoRepository.staffFindVideoByKeywords(keywords,pageable);

}

/\*\*

\* 分页分类返回所有视频条目

\* @param page

\* @param size

\* @return

\*/

public Page<Video> findAllClass(String keywords, int page,int size, String active, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page,size, direction, "create\_at");

Boolean activeTemp = Boolean.parseBoolean(active);

return videoRepository.findAllClassVideo(keywords, pageable, activeTemp);

}

/\*\*

\* 分页不分类返回所有视频条目

\* @param page

\* @param size

\* @return

\*/

public Page<Video> findAllExist(String keywords, int page, int size, Sort.Direction direction){

page--;

Pageable pageable = PageRequest.of(page, size, direction, "create\_at");

return videoRepository.findAllExistVideo(keywords, pageable);

}

/\*\*

\* 通过id找视频

\* @param id

\* @return

\*/

public Video findOneById(String id){

return videoRepository.findVideoById(id);

}

/\*\*

\* 更新视频

\* @param video

\*/

public void update(Video video){

Video videoTemp=this.findOneById(video.getId());

BeanUtils.copyProperties(video, videoTemp);

videoTemp.setUpdateAt(new Date());

videoTemp.setUpdateBy((String) session.getAttribute("user"));

videoRepository.saveAndFlush(videoTemp);

}

/\*\*

\* 删除一个视频

\* @param id

\*/

public void deleteOne(String id){

videoRepository.deleteById(id);

}

}

package com.miguo.matrix.config;

import com.miguo.matrix.interceptor.UserInterceptor;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.boot.SpringBootConfiguration;

import org.springframework.context.annotation.Bean;

import org.springframework.http.converter.json.MappingJackson2HttpMessageConverter;

import org.springframework.web.accept.ContentNegotiationManager;

import org.springframework.web.servlet.config.annotation.\*;

import org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter;

import java.util.Collections;

/\*\*

\* 功能描述：

\*

\* @author Hocassian

\* @date 2019-11-26 11:03

\*/

@EnableWebMvc

@SpringBootConfiguration

public class WebMvcConfig implements WebMvcConfigurer {

@Autowired

private UserInterceptor userInterceptor;

@Value("${file.path}")

private String filePath;

private static final String WINDOWS = "win";

@Bean

public RequestMappingHandlerAdapter requestMappingHandlerAdapter(@Autowired MappingJackson2HttpMessageConverter mappingJackson2HttpMessageConverter, @Autowired ContentNegotiationManager mvcContentNegotiationManager) {

RequestMappingHandlerAdapter requestMappingHandlerAdapter = new RequestMappingHandlerAdapter();

requestMappingHandlerAdapter.setMessageConverters(Collections.singletonList(mappingJackson2HttpMessageConverter));

requestMappingHandlerAdapter.setContentNegotiationManager(mvcContentNegotiationManager);

return requestMappingHandlerAdapter;

}

@Bean

public MappingJackson2HttpMessageConverter mappingJackson2HttpMessageConverter() {

return new MappingJackson2HttpMessageConverter();

}

@Override

public void addViewControllers(ViewControllerRegistry registry) {

}

@Override

public void addInterceptors(InterceptorRegistry registry){

registry.addInterceptor(userInterceptor).addPathPatterns("/staff/\*\*");

registry.addInterceptor(userInterceptor).addPathPatterns("/staff\_miniprogram/\*\*");

registry.addInterceptor(userInterceptor).addPathPatterns("/admin/\*\*");

}

@Override

public void addResourceHandlers(ResourceHandlerRegistry registry) {

registry.addResourceHandler("/static/\*\*").addResourceLocations("classpath:/static/");

registry.addResourceHandler("swagger-ui.html").addResourceLocations("classpath:/META-INF/resources/");

registry.addResourceHandler("/webjars/\*\*").addResourceLocations("classpath:/META-INF/resources/webjars/");

/\* 区分系统环境、自定义上传映射路径 \*/

String os = System.getProperty("os.name");

// 如果是Windows系统

if (os.toLowerCase().startsWith(WINDOWS)) {

registry.addResourceHandler("/file/\*\*")

.addResourceLocations("file:D:/File/test/");

} else { // linux 和mac

registry.addResourceHandler("/file/\*\*")

.addResourceLocations("file:./file/");

}

}

@Override

public void addCorsMappings(CorsRegistry registry) {

// 设置允许跨域的路径

registry.addMapping("/\*\*")

// 设置允许跨域请求的域名

.allowedOrigins("\*")

// 设置允许证书，不再默认开启

.allowCredentials(true)

// 设置允许的方法

.allowedMethods("GET", "POST", "PUT", "DELETE")

// 设置跨域允许时间

.maxAge(60 \* 60);

}

}

小程序前端代码

//app.js

App({

onLaunch: function () {

// 展示本地存储能力

var logs = wx.getStorageSync('logs') ||

logs.unshift(Date.now())

wx.setStorageSync('logs', logs)

},

globalData: {

userInfo: null,

activityId: '',

activityName: '',

activityProfile: '',

callbackUrl: 'https://migotimes.com/file/',

mpCallbackUrl: 'https://migotimes.com/file/thumbnails/mp',

serverAddr: 'https://migotimes.com/client\_miniprogram/',

},

/\*\*

\* 校验缓存

\*/

checkCache: function () {

if (wx.getStorageSync("session\_key") && wx.getStorageSync("openid") && this.globalData.userInfo) {

return true

} else {

return false

}

},

/\*\*

\* 微信用户登录

\*/

login: function () {

var that = this

if (this.checkCache()) {

return

}

wx.showLoading({

title: '正在获取用户凭证',

mask: true

})

// 登录

wx.login({

success: res => {

// 发送 res.code 到后台换取 openId, sessionKey, unionId

if (res.code) {

wx.setStorageSync('wxCode', res.code);

var \_url = this.globalData.serverAddr

wx.request({

url: \_url + 'user/login',

method: 'POST',

data: {

code: res.code

},

header: {

'content-type': 'application/x-www-form-urlencoded'

},

success(res) {

// console.log(res.data.data.openid);

if (res.data.openid != "" || res.data.data.openid != null) {

// 登录成功

wx.setStorageSync("openid", res.data.data.openid);//将用户id保存到缓存中

wx.setStorageSync("session\_key", res.data.data.session\_key);//将session\_key保存到缓存中

console.log('获取授权成功');

}

}

})

}

},

fail: function () {

wx.removeStorageSync(that.globalData.openid)

wx.showToast({

title: '获取用户凭证失败',

icon: 'none',

duration: 3000

})

},

timeout: function () {

wx.removeStorageSync(that.globalData.openid)

wx.showToast({

title: '获取用户凭证超时',

icon: 'none',

duration: 3000

})

}

})

wx.getSetting({

success: res => {

// console.log(res)

if (res.authSetting) {

// 已经授权，可以直接调用 getUserInfo 获取头像昵称，不会弹框

wx.getUserInfo({

success: res => {

// 可以将 res 发送给后台解码出 unionId

this.globalData.userInfo = res.userInfo

// 由于 getUserInfo 是网络请求，可能会在 Page.onLoad 之后才返回

// 所以此处加入 callback 以防止这种情况

if (this.userInfoReadyCallback) {

this.userInfoReadyCallback(res)

}

}

})

}

}

})

wx.hideLoading()

},

})

/\*\*app.wxss\*\*/

@import "style/fontawesome.wxss";

.container {

display: flex;

flex-direction: column;

align-items: center;

box-sizing: border-box;

}

button {

background: initial;

}

button:focus{

outline: 0;

}

button::after{

border: none;

}

page {

/\* background: linear-gradient(#def172,rgb(89, 189, 255)); \*/

/\* background: #f6f6f6; \*/

background-image: url("https://migotimes.com/file/mp/bg.png");

background-size: cover;

-webkit-background-size: cover;

-o-background-size: cover;

display: flex;

flex-direction: column;

justify-content: flex-start;

}

.userinfo, .uploader, .tunnel {

margin-top: 40rpx;

height: 140rpx;

width: 100%;

background: #fff;

border: 1px solid rgba(0, 0, 0, 0.1);

border-left: none;

border-right: none;

display: flex;

flex-direction: row;

align-items: center;

transition: all 300ms ease;

}

.userinfo-avatar {

width: 100rpx;

height: 100rpx;

margin: 20rpx;

border-radius: 50%;

background-size: cover;

background-color: white;

}

.userinfo-avatar:after {

border: none;

}

.userinfo-nickname {

font-size: 32rpx;

color: #007aff;

background-color: white;

background-size: cover;

}

.userinfo-nickname::after {

border: none;

}

.uploader, .tunnel {

height: auto;

padding: 0 0 0 40rpx;

flex-direction: column;

align-items: flex-start;

box-sizing: border-box;

}

.uploader-text, .tunnel-text {

width: 100%;

line-height: 52px;

font-size: 34rpx;

color: #007aff;

}

.uploader-container {

width: 100%;

height: 400rpx;

padding: 20rpx 20rpx 20rpx 0;

display: flex;

align-content: center;

justify-content: center;

box-sizing: border-box;

border-top: 1px solid rgba(0, 0, 0, 0.1);

}

.uploader-image {

width: 100%;

height: 360rpx;

}

.tunnel {

padding: 0 0 0 40rpx;

}

.tunnel-text {

position: relative;

color: #222;

display: flex;

flex-direction: row;

align-content: center;

justify-content: space-between;

box-sizing: border-box;

border-top: 1px solid rgba(0, 0, 0, 0.1);

}

.tunnel-text:first-child {

border-top: none;

}

.tunnel-switch {

position: absolute;

right: 20rpx;

top: -2rpx;

}

.disable {

color: #888;

}

.service {

position: fixed;

right: 40rpx;

bottom: 40rpx;

width: 140rpx;

height: 140rpx;

border-radius: 50%;

background: linear-gradient(#007aff, #0063ce);

box-shadow: 0 5px 10px rgba(0, 0, 0, 0.3);

display: flex;

align-content: center;

justify-content: center;

transition: all 300ms ease;

}

.service-button {

position: absolute;

top: 40rpx;

}

.service:active {

box-shadow: none;

}

.request-text {

padding: 20rpx 0;

font-size: 24rpx;

line-height: 36rpx;

word-break: break-all;

}

const FATAL\_REBUILD\_TOLERANCE = 10

const SETDATA\_SCROLL\_TO\_BOTTOM = {

scrollTop: 100000,

scrollWithAnimation: true,

}

Component({

properties: {

envId: String,

collection: String,

groupId: String,

groupName: String,

userInfo: Object,

onGetUserInfo: {

type: Function,

},

getOpenID: {

type: Function,

},

},

data: {

chats: ,

textInputValue: '',

openId: '',

scrollTop: 0,

scrollToMessage: '',

hasKeyboard: false,

},

methods: {

onGetUserInfo(e) {

this.properties.onGetUserInfo(e)

},

getOpenID() {

return this.properties.getOpenID()

},

mergeCommonCriteria(criteria) {

return {

groupId: this.data.groupId,

...criteria,

}

},

async initRoom() {

this.try(async () => {

await this.initOpenID()

const { envId, collection } = this.properties

const db = this.db = wx.cloud.database({

env: envId,

})

const \_ = db.command

const { data: initList } = await db.collection(collection).where(this.mergeCommonCriteria()).orderBy('sendTimeTS', 'desc').get()

console.log('init query chats', initList)

this.setData({

chats: initList.reverse(),

scrollTop: 10000,

})

this.initWatch(initList.length ? {

sendTimeTS: \_.gt(initList.sendTimeTS),

} : {})

}, '初始化失败')

},

async initOpenID() {

return this.try(async () => {

const openId = await this.getOpenID()

this.setData({

openId,

})

}, '初始化 openId 失败')

},

async initWatch(criteria) {

this.try(() => {

const { collection } = this.properties

const db = this.db

const \_ = db.command

console.warn(`开始监听`, criteria)

this.messageListener = db.collection(collection).where(this.mergeCommonCriteria(criteria)).watch({

onChange: this.onRealtimeMessageSnapshot.bind(this),

onError: e => {

if (!this.inited || this.fatalRebuildCount >= FATAL\_REBUILD\_TOLERANCE) {

this.showError(this.inited ? '监听错误，已断开' : '初始化监听失败', e, '重连', () => {

this.initWatch(this.data.chats.length ? {

sendTimeTS: \_.gt(this.data.chats.sendTimeTS),

} : {})

})

} else {

this.initWatch(this.data.chats.length ? {

sendTimeTS: \_.gt(this.data.chats.sendTimeTS),

} : {})

}

},

})

}, '初始化监听失败')

},

onRealtimeMessageSnapshot(snapshot) {

console.warn(`收到消息`, snapshot)

if (snapshot.type === 'init') {

this.setData({

chats: [

...this.data.chats,

....sort((x, y) => x.sendTimeTS - y.sendTimeTS),

],

})

this.scrollToBottom()

this.inited = true

} else {

let hasNewMessage = false

let hasOthersMessage = false

const chats =

for (const docChange of snapshot.docChanges) {

switch (docChange.queueType) {

case 'enqueue': {

hasOthersMessage = docChange.doc.\_openid !== this.data.openId

const ind = chats.findIndex(chat => chat.\_id === docChange.doc.\_id)

if (ind > -1) {

if (chats.msgType === 'image' && chats.tempFilePath) {

chats.splice(ind, 1, {

...docChange.doc,

tempFilePath: chats.tempFilePath,

})

} else chats.splice(ind, 1, docChange.doc)

} else {

hasNewMessage = true

chats.push(docChange.doc)

}

break

}

}

}

this.setData({

chats: chats.sort((x, y) => x.sendTimeTS - y.sendTimeTS),

})

if (hasOthersMessage || hasNewMessage) {

this.scrollToBottom()

}

}

},

async onConfirmSendText(e) {

this.try(async () => {

if (!e.detail.value) {

return

}

const { collection } = this.properties

const db = this.db

const \_ = db.command

const doc = {

\_id: `${Math.random()}\_${Date.now()}`,

groupId: this.data.groupId,

avatar: this.data.userInfo.avatarUrl,

nickName: this.data.userInfo.nickName,

msgType: 'text',

textContent: e.detail.value,

sendTime: new Date(),

sendTimeTS: Date.now(), // fallback

}

this.setData({

textInputValue: '',

chats: [

...this.data.chats,

{

...doc,

\_openid: this.data.openId,

writeStatus: 'pending',

},

],

})

this.scrollToBottom(true)

await db.collection(collection).add({

data: doc,

})

this.setData({

chats: this.data.chats.map(chat => {

if (chat.\_id === doc.\_id) {

return {

...chat,

writeStatus: 'written',

}

} else return chat

}),

})

}, '发送文字失败')

},

async onChooseImage(e) {

wx.chooseImage({

count: 1,

sourceType: ,

success: async res => {

const { envId, collection } = this.properties

const doc = {

\_id: `${Math.random()}\_${Date.now()}`,

groupId: this.data.groupId,

avatar: this.data.userInfo.avatarUrl,

nickName: this.data.userInfo.nickName,

msgType: 'image',

sendTime: new Date(),

sendTimeTS: Date.now(), // fallback

}

this.setData({

chats: [

...this.data.chats,

{

...doc,

\_openid: this.data.openId,

tempFilePath: res.tempFilePaths,

writeStatus: 0,

},

]

})

this.scrollToBottom(true)

const uploadTask = wx.cloud.uploadFile({

cloudPath: `${this.data.openId}/${Math.random()}\_${Date.now()}.${res.tempFilePaths.match(/\.(\w+)$/)}`,

filePath: res.tempFilePaths,

config: {

env: envId,

},

success: res => {

this.try(async () => {

await this.db.collection(collection).add({

data: {

...doc,

imgFileID: res.fileID,

},

})

}, '发送图片失败')

},

fail: e => {

this.showError('发送图片失败', e)

},

})

uploadTask.onProgressUpdate(({ progress }) => {

this.setData({

chats: this.data.chats.map(chat => {

if (chat.\_id === doc.\_id) {

return {

...chat,

writeStatus: progress,

}

} else return chat

})

})

})

},

})

},

onMessageImageTap(e) {

wx.previewImage({

urls: ,

})

},

scrollToBottom(force) {

if (force) {

console.log('force scroll to bottom')

this.setData(SETDATA\_SCROLL\_TO\_BOTTOM)

return

}

this.createSelectorQuery().select('.body').boundingClientRect(bodyRect => {

this.createSelectorQuery().select(`.body`).scrollOffset(scroll => {

if (scroll.scrollTop + bodyRect.height \* 3 > scroll.scrollHeight) {

console.log('should scroll to bottom')

this.setData(SETDATA\_SCROLL\_TO\_BOTTOM)

}

}).exec()

}).exec()

},

async onScrollToUpper() {

if (this.db && this.data.chats.length) {

const { collection } = this.properties

const \_ = this.db.command

const { data } = await this.db.collection(collection).where(this.mergeCommonCriteria({

sendTimeTS: \_.lt(this.data.chats.sendTimeTS),

})).orderBy('sendTimeTS', 'desc').get()

this.data.chats.unshift(...data.reverse())

this.setData({

chats: this.data.chats,

scrollToMessage: `item-${data.length}`,

scrollWithAnimation: false,

})

}

},

async try(fn, title) {

try {

await fn()

} catch (e) {

this.showError(title, e)

}

},

showError(title, content, confirmText, confirmCallback) {

console.error(title, content)

wx.showModal({

title,

content: content.toString(),

showCancel: confirmText ? true : false,

confirmText,

success: res => {

res.confirm && confirmCallback()

},

})

},

},

ready() {

global.chatroom = this

this.initRoom()

this.fatalRebuildCount = 0

},

})

<view class="chatroom">

<!-- chats -->

<scroll-view

class="body"

scroll-y

scroll-with-animation="{{scrollWithAnimation}}"

scroll-top="{{scrollTop}}"

scroll-into-view="{{scrollToMessage}}"

bindscrolltoupper="onScrollToUpper"

>

<view

wx:for="{{chats}}"

wx:key="{{item.\_id}}"

id="item-{{index}}"

class="message {{openId == item.\_openid ? 'message\_\_self' : ''}}"

>

<image

class="avatar"

src="{{item.avatar}}"

mode="scaleToFill"

></image>

<view class="main">

<view class="nickname">{{item.nickName}}</view>

<block wx:if="{{item.msgType === 'image'}}">

<view class="image-wrapper">

<view class="loading" wx:if="{{item.writeStatus > -1}}">{{item.writeStatus}}%</view>

<image

src="{{item.tempFilePath || item.imgFileID}}"

data-fileid="{{item.tempFilePath || item.imgFileID}}"

class="image-content"

style="{{item.imgStyle}}"

mode="scallToFill"

bindtap="onMessageImageTap"></image>

</view>

</block>

<block wx:else>

<view class="text-wrapper">

<view class="loading" wx:if="{{item.writeStatus === 'pending'}}">···</view>

<view class="text-content">{{item.textContent}}</view>

</view>

</block>

</view>

</view>

</scroll-view>

<!-- message sender -->

<view class="footer">

<view class="message-sender" wx:if="{{userInfo}}">

<input

class="text-input"

type="text"

confirm-type="send"

bindconfirm="onConfirmSendText"

cursor-spacing="20"

value="{{textInputValue}}"

></input>

<image

src="./photo.png"

class="btn-send-image"

mode="scaleToFill"

bindtap="onChooseImage"

></image>

</view>

<view class="message-sender" wx:if="{{!userInfo}}">

<button

open-type="getUserInfo"

bindgetuserinfo="onGetUserInfo"

class="userinfo"

>请先登录后参与聊天</button>

</view>

</view>

</view>

.chatroom {

width: 100%;

height: 100%;

display: flex;

flex-direction: column;

}

.chatroom .header {

flex-basis: fit-content;

display: flex;

flex-direction: row;

border-bottom: 1px solid #ddd;

padding: 20rpx 0 30rpx;

font-size: 30rpx;

/\* background: rgb(34, 187, 47);

color: rgba(255, 255, 255, 1) \*/

/\* font-family: 'Microsoft YaHei' \*/

}

.chatroom .header .left {

flex: 1;

}

.chatroom .header .middle {

flex: 2;

text-align: center;

}

.chatroom .header .right {

flex: 1;

}

.chatroom .body {

flex: 2;

display: flex;

flex-direction: column;

background: rgb(237,237,237);

padding-bottom: 16rpx;

}

.body .message {

display: flex;

flex-direction: row;

position: relative;

margin: 12rpx 0;

}

.body .message.message\_\_self {

flex-direction: row-reverse;

}

.body .message .avatar {

position: relative;

top: 5rpx;

width: 60rpx;

height: 60rpx;

border-radius: 5rpx;

margin: 15rpx;

}

.body .message .main {

flex: 1;

display: flex;

flex-direction: column;

align-items: flex-start;

}

.body .message.message\_\_self .main {

align-items: flex-end;

}

.body .message .nickname {

font-size: 24rpx;

color: #444;

}

.body .message .text-content {

border: 1px solid transparent;

border-radius: 3px;

background-color: #fff;

margin: 2px 0 0 0;

padding: 4px 10px;

font-size: 30rpx;

display: inline-block;

}

.body .message.message\_\_self .text-content {

background-color: paleturquoise;

}

.body .message .text-wrapper {

display: flex;

flex-direction: row;

align-items: center;

max-width: 80%;

}

.body .message.message\_\_self .text-wrapper .loading{

font-size: 16rpx;

margin-right: 18rpx;

}

.body .message .image-wrapper {

display: flex;

flex-direction: row;

align-items: center;

}

.body .message .image-content {

max-width: 240rpx;

max-height: 240rpx;

}

.body .message.message\_\_self .image-wrapper .loading {

font-size: 20rpx;

margin-right: 18rpx;

}

.chatroom .footer {

flex-basis: fit-content;

display: flex;

flex-direction: row;

border-top: 1px solid #ddd;

font-size: 10rpx;

padding: 20rpx 30rpx;

background: rgb(246,246,246);

}

.chatroom .footer .message-sender {

flex: 1;

display: flex;

flex-direction: row;

}

.message-sender .text-input {

flex: 1;

font-size: 16px;

border: 1px solid transparent;

border-radius: 5px;

padding: 3px 6px;

margin: 0 10px 0 5px;

background: #fff;

}

.message-sender .btn-send-image {

width: 50rpx;

height: 50rpx;

align-self: center;

}

button {

font-size: 30rpx;

}

button.userinfo {

background: darkturquoise;

color: aliceblue;

padding: 0 100rpx;

border: 1px solid #ddd;

border-radius: 20px;

}

const app = getApp()

function getAPI(x\_method, data, x\_url) {

return new Promise(function (resolve, reject) {

var f\_url = app.globalData.serverAddr + x\_url

wx.request({

url: f\_url,

method: x\_method,

data: data,

success: function (res) {

resolve(res);

},

fail: () => {

reject("系统异常，请重试！")

}

})

})

}

module.exports = {

getAPI: getAPI

}

/\*!

\* Font Awesome 4.7.0 by @davegandy - http://fontawesome.io - @fontawesome

\* License - http://fontawesome.io/license (Font: SIL OFL 1.1, CSS: MIT License)

\*/

/\* FONT PATH

\* -------------------------- \*/

@font-face {

font-family: 'FontAwesome';

src: url('https://cdn.staticfile.org/font-awesome/4.7.0/css/font-awesome.css');

src:

url('https://cdn.staticfile.org/font-awesome/4.7.0/fonts/fontawesome-webfont.woff2') format('woff2'),

url('https://cdn.staticfile.org/font-awesome/4.7.0/fonts/fontawesome-webfont.woff') format('woff'),

url('https://cdn.staticfile.org/font-awesome/4.7.0/fonts/fontawesome-webfont.svg') format('svg'),

url('https://cdn.staticfile.org/font-awesome/4.7.0/fonts/fontawesome-webfont.eot') format('eot'),

url('https://cdn.staticfile.org/font-awesome/4.7.0/fonts/fontawesome-webfont.ttf') format('ttf');

font-weight: normal;

font-style: normal;

}

.fa {

display: inline-block;

font: normal normal normal 14px/1 FontAwesome;

font-size: inherit;

text-rendering: auto;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

}

/\* makes the font 33% larger relative to the icon container \*/

.fa-lg {

font-size: 1.33333333em;

line-height: 0.75em;

vertical-align: -15%;

}

.fa-2x {

font-size: 2em;

}

.fa-3x {

font-size: 3em;

}

.fa-4x {

font-size: 4em;

}

.fa-5x {

font-size: 5em;

}

.fa-fw {

width: 1.28571429em;

text-align: center;

}

.fa-ul {

padding-left: 0;

margin-left: 2.14285714em;

list-style-type: none;

}

.fa-ul > li {

position: relative;

}

.fa-li {

position: absolute;

left: -2.14285714em;

width: 2.14285714em;

top: 0.14285714em;

text-align: center;

}

.fa-li.fa-lg {

left: -1.85714286em;

}

.fa-border {

padding: .2em .25em .15em;

border: solid 0.08em #eeeeee;

border-radius: .1em;

}

.fa-pull-left {

float: left;

}

.fa-pull-right {

float: right;

}

.fa.fa-pull-left {

margin-right: .3em;

}

.fa.fa-pull-right {

margin-left: .3em;

}

/\* Deprecated as of 4.4.0 \*/

.pull-right {

float: right;

}

.pull-left {

float: left;

}

.fa.pull-left {

margin-right: .3em;

}

.fa.pull-right {

margin-left: .3em;

}

.fa-spin {

-webkit-animation: fa-spin 2s infinite linear;

animation: fa-spin 2s infinite linear;

}

.fa-pulse {

-webkit-animation: fa-spin 1s infinite steps(8);

animation: fa-spin 1s infinite steps(8);

}

@-webkit-keyframes fa-spin {

0% {

-webkit-transform: rotate(0deg);

transform: rotate(0deg);

}

100% {

-webkit-transform: rotate(359deg);

transform: rotate(359deg);

}

}

@keyframes fa-spin {

0% {

-webkit-transform: rotate(0deg);

transform: rotate(0deg);

}

100% {

-webkit-transform: rotate(359deg);

transform: rotate(359deg);

}

}

.fa-rotate-90 {

-ms-filter: "progid:DXImageTransform.Microsoft.BasicImage(rotation=1)";

-webkit-transform: rotate(90deg);

-ms-transform: rotate(90deg);

transform: rotate(90deg);

}

.fa-rotate-180 {

-ms-filter: "progid:DXImageTransform.Microsoft.BasicImage(rotation=2)";

-webkit-transform: rotate(180deg);

-ms-transform: rotate(180deg);

transform: rotate(180deg);

}

.fa-rotate-270 {

-ms-filter: "progid:DXImageTransform.Microsoft.BasicImage(rotation=3)";

-webkit-transform: rotate(270deg);

-ms-transform: rotate(270deg);

transform: rotate(270deg);

}

.fa-flip-horizontal {

-ms-filter: "progid:DXImageTransform.Microsoft.BasicImage(rotation=0, mirror=1)";

-webkit-transform: scale(-1, 1);

-ms-transform: scale(-1, 1);

transform: scale(-1, 1);

}

.fa-flip-vertical {

-ms-filter: "progid:DXImageTransform.Microsoft.BasicImage(rotation=2, mirror=1)";

-webkit-transform: scale(1, -1);

-ms-transform: scale(1, -1);

transform: scale(1, -1);

}

:root .fa-rotate-90,

:root .fa-rotate-180,

:root .fa-rotate-270,

:root .fa-flip-horizontal,

:root .fa-flip-vertical {

filter: none;

}

.fa-stack {

position: relative;

display: inline-block;

width: 2em;

height: 2em;

line-height: 2em;

vertical-align: middle;

}

.fa-stack-1x,

.fa-stack-2x {

position: absolute;

left: 0;

width: 100%;

text-align: center;

}

.fa-stack-1x {

line-height: inherit;

}

.fa-stack-2x {

font-size: 2em;

}

.fa-inverse {

color: #ffffff;

}

/\* Font Awesome uses the Unicode Private Use Area (PUA) to ensure screen

readers do not read off random characters that represent icons \*/

.fa-glass:before {

content: "\f000";

}

.fa-music:before {

content: "\f001";

}

.fa-search:before {

content: "\f002";

}

.fa-envelope-o:before {

content: "\f003";

}

.fa-heart:before {

content: "\f004";

}

.fa-star:before {

content: "\f005";

}

.fa-star-o:before {

content: "\f006";

}

.fa-user:before {

content: "\f007";

}

.fa-film:before {

content: "\f008";

}

.fa-th-large:before {

content: "\f009";

}

.fa-th:before {

content: "\f00a";

}

.fa-th-list:before {

content: "\f00b";

}

.fa-check:before {

content: "\f00c";

}

.fa-remove:before,

.fa-close:before,

.fa-times:before {

content: "\f00d";

}

.fa-search-plus:before {

content: "\f00e";

}

.fa-search-minus:before {

content: "\f010";

}

.fa-power-off:before {

content: "\f011";

}

.fa-signal:before {

content: "\f012";

}

.fa-gear:before,

.fa-cog:before {

content: "\f013";

}

.fa-trash-o:before {

content: "\f014";

}

.fa-home:before {

content: "\f015";

}

.fa-file-o:before {

content: "\f016";

}

.fa-clock-o:before {

content: "\f017";

}

.fa-road:before {

content: "\f018";

}

.fa-download:before {

content: "\f019";

}

.fa-arrow-circle-o-down:before {

content: "\f01a";

}

.fa-arrow-circle-o-up:before {

content: "\f01b";

}

.fa-inbox:before {

content: "\f01c";

}

.fa-play-circle-o:before {

content: "\f01d";

}

.fa-rotate-right:before,

.fa-repeat:before {

content: "\f01e";

}

.fa-refresh:before {

content: "\f021";

}

.fa-list-alt:before {

content: "\f022";

}

.fa-lock:before {

content: "\f023";

}

.fa-flag:before {

content: "\f024";

}

.fa-headphones:before {

content: "\f025";

}

.fa-volume-off:before {

content: "\f026";

}

.fa-volume-down:before {

content: "\f027";

}

.fa-volume-up:before {

content: "\f028";

}

.fa-qrcode:before {

content: "\f029";

}

.fa-barcode:before {

content: "\f02a";

}

.fa-tag:before {

content: "\f02b";

}

.fa-tags:before {

content: "\f02c";

}

.fa-book:before {

content: "\f02d";

}

.fa-bookmark:before {

content: "\f02e";

}

.fa-print:before {

content: "\f02f";

}

.fa-camera:before {

content: "\f030";

}

.fa-font:before {

content: "\f031";

}

.fa-bold:before {

content: "\f032";

}

.fa-italic:before {

content: "\f033";

}

.fa-text-height:before {

content: "\f034";

}

.fa-text-width:before {

content: "\f035";

}

.fa-align-left:before {

content: "\f036";

}

.fa-align-center:before {

content: "\f037";

}

.fa-align-right:before {

content: "\f038";

}

.fa-align-justify:before {

content: "\f039";

}

.fa-list:before {

content: "\f03a";

}

.fa-dedent:before,

.fa-outdent:before {

content: "\f03b";

}

.fa-indent:before {

content: "\f03c";

}

.fa-video-camera:before {

content: "\f03d";

}

.fa-photo:before,

.fa-image:before,

.fa-picture-o:before {

content: "\f03e";

}

.fa-pencil:before {

content: "\f040";

}

.fa-map-marker:before {

content: "\f041";

}

.fa-adjust:before {

content: "\f042";

}

.fa-tint:before {

content: "\f043";

}

.fa-edit:before,

.fa-pencil-square-o:before {

content: "\f044";

}

.fa-share-square-o:before {

content: "\f045";

}

.fa-check-square-o:before {

content: "\f046";

}

.fa-arrows:before {

content: "\f047";

}

.fa-step-backward:before {

content: "\f048";

}

.fa-fast-backward:before {

content: "\f049";

}

.fa-backward:before {

content: "\f04a";

}

.fa-play:before {

content: "\f04b";

}

.fa-pause:before {

content: "\f04c";

}

.fa-stop:before {

content: "\f04d";

}

.fa-forward:before {

content: "\f04e";

}

.fa-fast-forward:before {

content: "\f050";

}

.fa-step-forward:before {

content: "\f051";

}

.fa-eject:before {

content: "\f052";

}

.fa-chevron-left:before {

content: "\f053";

}

.fa-chevron-right:before {

content: "\f054";

}

.fa-plus-circle:before {

content: "\f055";

}

.fa-minus-circle:before {

content: "\f056";

}

.fa-times-circle:before {

content: "\f057";

}

.fa-check-circle:before {

content: "\f058";

}

.fa-question-circle:before {

content: "\f059";

}

.fa-info-circle:before {

content: "\f05a";

}

.fa-crosshairs:before {

content: "\f05b";

}

.fa-times-circle-o:before {

content: "\f05c";

}

.fa-check-circle-o:before {

content: "\f05d";

}

.fa-ban:before {

content: "\f05e";

}

.fa-arrow-left:before {

content: "\f060";

}

.fa-arrow-right:before {

content: "\f061";

}

.fa-arrow-up:before {

content: "\f062";

}

.fa-arrow-down:before {

content: "\f063";

}

.fa-mail-forward:before,

.fa-share:before {

content: "\f064";

}

.fa-expand:before {

content: "\f065";

}

.fa-compress:before {

content: "\f066";

}

.fa-plus:before {

content: "\f067";

}

.fa-minus:before {

content: "\f068";

}

.fa-asterisk:before {

content: "\f069";

}

.fa-exclamation-circle:before {

content: "\f06a";

}

.fa-gift:before {

content: "\f06b";

}

.fa-leaf:before {

content: "\f06c";

}

.fa-fire:before {

content: "\f06d";

}

.fa-eye:before {

content: "\f06e";

}

.fa-eye-slash:before {

content: "\f070";

}

.fa-warning:before,

.fa-exclamation-triangle:before {

content: "\f071";

}

.fa-plane:before {

content: "\f072";

}

.fa-calendar:before {

content: "\f073";

}

.fa-random:before {

content: "\f074";

}

.fa-comment:before {

content: "\f075";

}

.fa-magnet:before {

content: "\f076";

}

.fa-chevron-up:before {

content: "\f077";

}

.fa-chevron-down:before {

content: "\f078";

}

.fa-retweet:before {

content: "\f079";

}

.fa-shopping-cart:before {

content: "\f07a";

}

.fa-folder:before {

content: "\f07b";

}

.fa-folder-open:before {

content: "\f07c";

}

.fa-arrows-v:before {

content: "\f07d";

}

.fa-arrows-h:before {

content: "\f07e";

}

.fa-bar-chart-o:before,

.fa-bar-chart:before {

content: "\f080";

}

.fa-twitter-square:before {

content: "\f081";

}

.fa-facebook-square:before {

content: "\f082";

}

.fa-camera-retro:before {

content: "\f083";

}

.fa-key:before {

content: "\f084";

}

.fa-gears:before,

.fa-cogs:before {

content: "\f085";

}

.fa-comments:before {

content: "\f086";

}

.fa-thumbs-o-up:before {

content: "\f087";

}

.fa-thumbs-o-down:before {

content: "\f088";

}

.fa-star-half:before {

content: "\f089";

}

.fa-heart-o:before {

content: "\f08a";

}

.fa-sign-out:before {

content: "\f08b";

}

.fa-linkedin-square:before {

content: "\f08c";

}

.fa-thumb-tack:before {

content: "\f08d";

}

.fa-external-link:before {

content: "\f08e";

}

.fa-sign-in:before {

content: "\f090";

}

.fa-trophy:before {

content: "\f091";

}

.fa-github-square:before {

content: "\f092";

}

.fa-upload:before {

content: "\f093";

}

.fa-lemon-o:before {

content: "\f094";

}

.fa-phone:before {

content: "\f095";

}

.fa-square-o:before {

content: "\f096";

}

.fa-bookmark-o:before {

content: "\f097";

}

.fa-phone-square:before {

content: "\f098";

}

.fa-twitter:before {

content: "\f099";

}

.fa-facebook-f:before,

.fa-facebook:before {

content: "\f09a";

}

.fa-github:before {

content: "\f09b";

}

.fa-unlock:before {

content: "\f09c";

}

.fa-credit-card:before {

content: "\f09d";

}

.fa-feed:before,

.fa-rss:before {

content: "\f09e";

}

.fa-hdd-o:before {

content: "\f0a0";

}

.fa-bullhorn:before {

content: "\f0a1";

}

.fa-bell:before {

content: "\f0f3";

}

.fa-certificate:before {

content: "\f0a3";

}

.fa-hand-o-right:before {

content: "\f0a4";

}

.fa-hand-o-left:before {

content: "\f0a5";

}

.fa-hand-o-up:before {

content: "\f0a6";

}

.fa-hand-o-down:before {

content: "\f0a7";

}

.fa-arrow-circle-left:before {

content: "\f0a8";

}

.fa-arrow-circle-right:before {

content: "\f0a9";

}

.fa-arrow-circle-up:before {

content: "\f0aa";

}

.fa-arrow-circle-down:before {

content: "\f0ab";

}

.fa-globe:before {

content: "\f0ac";

}

.fa-wrench:before {

content: "\f0ad";

}

.fa-tasks:before {

content: "\f0ae";

}

.fa-filter:before {

content: "\f0b0";

}

.fa-briefcase:before {

content: "\f0b1";

}

.fa-arrows-alt:before {

content: "\f0b2";

}

.fa-group:before,

.fa-users:before {

content: "\f0c0";

}

.fa-chain:before,

.fa-link:before {

content: "\f0c1";

}

.fa-cloud:before {

content: "\f0c2";

}

.fa-flask:before {

content: "\f0c3";

}

.fa-cut:before,

.fa-scissors:before {

content: "\f0c4";

}

.fa-copy:before,

.fa-files-o:before {

content: "\f0c5";

}

.fa-paperclip:before {

content: "\f0c6";

}

.fa-save:before,

.fa-floppy-o:before {

content: "\f0c7";

}

.fa-square:before {

content: "\f0c8";

}

.fa-navicon:before,

.fa-reorder:before,

.fa-bars:before {

content: "\f0c9";

}

.fa-list-ul:before {

content: "\f0ca";

}

.fa-list-ol:before {

content: "\f0cb";

}

.fa-strikethrough:before {

content: "\f0cc";

}

.fa-underline:before {

content: "\f0cd";

}

.fa-table:before {

content: "\f0ce";

}

.fa-magic:before {

content: "\f0d0";

}

.fa-truck:before {

content: "\f0d1";

}

.fa-pinterest:before {

content: "\f0d2";

}

.fa-pinterest-square:before {

content: "\f0d3";

}

.fa-google-plus-square:before {

content: "\f0d4";

}

.fa-google-plus:before {

content: "\f0d5";

}

.fa-money:before {

content: "\f0d6";

}

.fa-caret-down:before {

content: "\f0d7";

}

.fa-caret-up:before {

content: "\f0d8";

}

.fa-caret-left:before {

content: "\f0d9";

}

.fa-caret-right:before {

content: "\f0da";

}

.fa-columns:before {

content: "\f0db";

}

.fa-unsorted:before,

.fa-sort:before {

content: "\f0dc";

}

.fa-sort-down:before,

.fa-sort-desc:before {

content: "\f0dd";

}

.fa-sort-up:before,

.fa-sort-asc:before {

content: "\f0de";

}

.fa-envelope:before {

content: "\f0e0";

}

.fa-linkedin:before {

content: "\f0e1";

}

.fa-rotate-left:before,

.fa-undo:before {

content: "\f0e2";

}

.fa-legal:before,

.fa-gavel:before {

content: "\f0e3";

}

.fa-dashboard:before,

.fa-tachometer:before {

content: "\f0e4";

}

.fa-comment-o:before {

content: "\f0e5";

}

.fa-comments-o:before {

content: "\f0e6";

}

.fa-flash:before,

.fa-bolt:before {

content: "\f0e7";

}

.fa-sitemap:before {

content: "\f0e8";

}

.fa-umbrella:before {

content: "\f0e9";

}

.fa-paste:before,

.fa-clipboard:before {

content: "\f0ea";

}

.fa-lightbulb-o:before {

content: "\f0eb";

}

.fa-exchange:before {

content: "\f0ec";

}

.fa-cloud-download:before {

content: "\f0ed";

}

.fa-cloud-upload:before {

content: "\f0ee";

}

.fa-user-md:before {

content: "\f0f0";

}

.fa-stethoscope:before {

content: "\f0f1";

}

.fa-suitcase:before {

content: "\f0f2";

}

.fa-bell-o:before {

content: "\f0a2";

}

.fa-coffee:before {

content: "\f0f4";

}

.fa-cutlery:before {

content: "\f0f5";

}

.fa-file-text-o:before {

content: "\f0f6";

}

.fa-building-o:before {

content: "\f0f7";

}

.fa-hospital-o:before {

content: "\f0f8";

}

.fa-ambulance:before {

content: "\f0f9";

}

.fa-medkit:before {

content: "\f0fa";

}

.fa-fighter-jet:before {

content: "\f0fb";

}

.fa-beer:before {

content: "\f0fc";

}

.fa-h-square:before {

content: "\f0fd";

}

.fa-plus-square:before {

content: "\f0fe";

}

.fa-angle-double-left:before {

content: "\f100";

}

.fa-angle-double-right:before {

content: "\f101";

}

.fa-angle-double-up:before {

content: "\f102";

}

.fa-angle-double-down:before {

content: "\f103";

}

.fa-angle-left:before {

content: "\f104";

}

.fa-angle-right:before {

content: "\f105";

}

.fa-angle-up:before {

content: "\f106";

}

.fa-angle-down:before {

content: "\f107";

}

.fa-desktop:before {

content: "\f108";

}

.fa-laptop:before {

content: "\f109";

}

.fa-tablet:before {

content: "\f10a";

}

.fa-mobile-phone:before,

.fa-mobile:before {

content: "\f10b";

}

.fa-circle-o:before {

content: "\f10c";

}

.fa-quote-left:before {

content: "\f10d";

}

.fa-quote-right:before {

content: "\f10e";

}

.fa-spinner:before {

content: "\f110";

}

.fa-circle:before {

content: "\f111";

}

.fa-mail-reply:before,

.fa-reply:before {

content: "\f112";

}

.fa-github-alt:before {

content: "\f113";

}

.fa-folder-o:before {

content: "\f114";

}

.fa-folder-open-o:before {

content: "\f115";

}

.fa-smile-o:before {

content: "\f118";

}

.fa-frown-o:before {

content: "\f119";

}

.fa-meh-o:before {

content: "\f11a";

}

.fa-gamepad:before {

content: "\f11b";

}

.fa-keyboard-o:before {

content: "\f11c";

}

.fa-flag-o:before {

content: "\f11d";

}

.fa-flag-checkered:before {

content: "\f11e";

}

.fa-terminal:before {

content: "\f120";

}

.fa-code:before {

content: "\f121";

}

.fa-mail-reply-all:before,

.fa-reply-all:before {

content: "\f122";

}

.fa-star-half-empty:before,

.fa-star-half-full:before,

.fa-star-half-o:before {

content: "\f123";

}

.fa-location-arrow:before {

content: "\f124";

}

.fa-crop:before {

content: "\f125";

}

.fa-code-fork:before {

content: "\f126";

}

.fa-unlink:before,

.fa-chain-broken:before {

content: "\f127";

}

.fa-question:before {

content: "\f128";

}

.fa-info:before {

content: "\f129";

}

.fa-exclamation:before {

content: "\f12a";

}

.fa-superscript:before {

content: "\f12b";

}

.fa-subscript:before {

content: "\f12c";

}

.fa-eraser:before {

content: "\f12d";

}

.fa-puzzle-piece:before {

content: "\f12e";

}

.fa-microphone:before {

content: "\f130";

}

.fa-microphone-slash:before {

content: "\f131";

}

.fa-shield:before {

content: "\f132";

}

.fa-calendar-o:before {

content: "\f133";

}

.fa-fire-extinguisher:before {

content: "\f134";

}

.fa-rocket:before {

content: "\f135";

}

.fa-maxcdn:before {

content: "\f136";

}

.fa-chevron-circle-left:before {

content: "\f137";

}

.fa-chevron-circle-right:before {

content: "\f138";

}

.fa-chevron-circle-up:before {

content: "\f139";

}

.fa-chevron-circle-down:before {

content: "\f13a";

}

.fa-html5:before {

content: "\f13b";

}

.fa-css3:before {

content: "\f13c";

}

.fa-anchor:before {

content: "\f13d";

}

.fa-unlock-alt:before {

content: "\f13e";

}

.fa-bullseye:before {

content: "\f140";

}

.fa-ellipsis-h:before {

content: "\f141";

}

.fa-ellipsis-v:before {

content: "\f142";

}

.fa-rss-square:before {

content: "\f143";

}

.fa-play-circle:before {

content: "\f144";

}

.fa-ticket:before {

content: "\f145";

}

.fa-minus-square:before {

content: "\f146";

}

.fa-minus-square-o:before {

content: "\f147";

}

.fa-level-up:before {

content: "\f148";

}

.fa-level-down:before {

content: "\f149";

}

.fa-check-square:before {

content: "\f14a";

}

.fa-pencil-square:before {

content: "\f14b";

}

.fa-external-link-square:before {

content: "\f14c";

}

.fa-share-square:before {

content: "\f14d";

}

.fa-compass:before {

content: "\f14e";

}

.fa-toggle-down:before,

.fa-caret-square-o-down:before {

content: "\f150";

}

.fa-toggle-up:before,

.fa-caret-square-o-up:before {

content: "\f151";

}

.fa-toggle-right:before,

.fa-caret-square-o-right:before {

content: "\f152";

}

.fa-euro:before,

.fa-eur:before {

content: "\f153";

}

.fa-gbp:before {

content: "\f154";

}

.fa-dollar:before,

.fa-usd:before {

content: "\f155";

}

.fa-rupee:before,

.fa-inr:before {

content: "\f156";

}

.fa-cny:before,

.fa-rmb:before,

.fa-yen:before,

.fa-jpy:before {

content: "\f157";

}

.fa-ruble:before,

.fa-rouble:before,

.fa-rub:before {

content: "\f158";

}

.fa-won:before,

.fa-krw:before {

content: "\f159";

}

.fa-bitcoin:before,

.fa-btc:before {

content: "\f15a";

}

.fa-file:before {

content: "\f15b";

}

.fa-file-text:before {

content: "\f15c";

}

.fa-sort-alpha-asc:before {

content: "\f15d";

}

.fa-sort-alpha-desc:before {

content: "\f15e";

}

.fa-sort-amount-asc:before {

content: "\f160";

}

.fa-sort-amount-desc:before {

content: "\f161";

}

.fa-sort-numeric-asc:before {

content: "\f162";

}

.fa-sort-numeric-desc:before {

content: "\f163";

}

.fa-thumbs-up:before {

content: "\f164";

}

.fa-thumbs-down:before {

content: "\f165";

}

.fa-youtube-square:before {

content: "\f166";

}

.fa-youtube:before {

content: "\f167";

}

.fa-xing:before {

content: "\f168";

}

.fa-xing-square:before {

content: "\f169";

}

.fa-youtube-play:before {

content: "\f16a";

}

.fa-dropbox:before {

content: "\f16b";

}

.fa-stack-overflow:before {

content: "\f16c";

}

.fa-instagram:before {

content: "\f16d";

}

.fa-flickr:before {

content: "\f16e";

}

.fa-adn:before {

content: "\f170";

}

.fa-bitbucket:before {

content: "\f171";

}

.fa-bitbucket-square:before {

content: "\f172";

}

.fa-tumblr:before {

content: "\f173";

}

.fa-tumblr-square:before {

content: "\f174";

}

.fa-long-arrow-down:before {

content: "\f175";

}

.fa-long-arrow-up:before {

content: "\f176";

}

.fa-long-arrow-left:before {

content: "\f177";

}

.fa-long-arrow-right:before {

content: "\f178";

}

.fa-apple:before {

content: "\f179";

}

.fa-windows:before {

content: "\f17a";

}

.fa-android:before {

content: "\f17b";

}

.fa-linux:before {

content: "\f17c";

}

.fa-dribbble:before {

content: "\f17d";

}

.fa-skype:before {

content: "\f17e";

}

.fa-foursquare:before {

content: "\f180";

}

.fa-trello:before {

content: "\f181";

}

.fa-female:before {

content: "\f182";

}

.fa-male:before {

content: "\f183";

}

.fa-gittip:before,

.fa-gratipay:before {

content: "\f184";

}

.fa-sun-o:before {

content: "\f185";

}

.fa-moon-o:before {

content: "\f186";

}

.fa-archive:before {

content: "\f187";

}

.fa-bug:before {

content: "\f188";

}

.fa-vk:before {

content: "\f189";

}

.fa-weibo:before {

content: "\f18a";

}

.fa-renren:before {

content: "\f18b";

}

.fa-pagelines:before {

content: "\f18c";

}

.fa-stack-exchange:before {

content: "\f18d";

}

.fa-arrow-circle-o-right:before {

content: "\f18e";

}

.fa-arrow-circle-o-left:before {

content: "\f190";

}

.fa-toggle-left:before,

.fa-caret-square-o-left:before {

content: "\f191";

}

.fa-dot-circle-o:before {

content: "\f192";

}

.fa-wheelchair:before {

content: "\f193";

}

.fa-vimeo-square:before {

content: "\f194";

}

.fa-turkish-lira:before,

.fa-try:before {

content: "\f195";

}

.fa-plus-square-o:before {

content: "\f196";

}

.fa-space-shuttle:before {

content: "\f197";

}

.fa-slack:before {

content: "\f198";

}

.fa-envelope-square:before {

content: "\f199";

}

.fa-wordpress:before {

content: "\f19a";

}

.fa-openid:before {

content: "\f19b";

}

.fa-institution:before,

.fa-bank:before,

.fa-university:before {

content: "\f19c";

}

.fa-mortar-board:before,

.fa-graduation-cap:before {

content: "\f19d";

}

.fa-yahoo:before {

content: "\f19e";

}

.fa-google:before {

content: "\f1a0";

}

.fa-reddit:before {

content: "\f1a1";

}

.fa-reddit-square:before {

content: "\f1a2";

}

.fa-stumbleupon-circle:before {

content: "\f1a3";

}

.fa-stumbleupon:before {

content: "\f1a4";

}

.fa-delicious:before {

content: "\f1a5";

}

.fa-digg:before {

content: "\f1a6";

}

.fa-pied-piper-pp:before {

content: "\f1a7";

}

.fa-pied-piper-alt:before {

content: "\f1a8";

}

.fa-drupal:before {

content: "\f1a9";

}

.fa-joomla:before {

content: "\f1aa";

}

.fa-language:before {

content: "\f1ab";

}

.fa-fax:before {

content: "\f1ac";

}

.fa-building:before {

content: "\f1ad";

}

.fa-child:before {

content: "\f1ae";

}

.fa-paw:before {

content: "\f1b0";

}

.fa-spoon:before {

content: "\f1b1";

}

.fa-cube:before {

content: "\f1b2";

}

.fa-cubes:before {

content: "\f1b3";

}

.fa-behance:before {

content: "\f1b4";

}

.fa-behance-square:before {

content: "\f1b5";

}

.fa-steam:before {

content: "\f1b6";

}

.fa-steam-square:before {

content: "\f1b7";

}

.fa-recycle:before {

content: "\f1b8";

}

.fa-automobile:before,

.fa-car:before {

content: "\f1b9";

}

.fa-cab:before,

.fa-taxi:before {

content: "\f1ba";

}

.fa-tree:before {

content: "\f1bb";

}

.fa-spotify:before {

content: "\f1bc";

}

.fa-deviantart:before {

content: "\f1bd";

}

.fa-soundcloud:before {

content: "\f1be";

}

.fa-database:before {

content: "\f1c0";

}

.fa-file-pdf-o:before {

content: "\f1c1";

}

.fa-file-word-o:before {

content: "\f1c2";

}

.fa-file-excel-o:before {

content: "\f1c3";

}

.fa-file-powerpoint-o:before {

content: "\f1c4";

}

.fa-file-photo-o:before,

.fa-file-picture-o:before,

.fa-file-image-o:before {

content: "\f1c5";

}

.fa-file-zip-o:before,

.fa-file-archive-o:before {

content: "\f1c6";

}

.fa-file-sound-o:before,

.fa-file-audio-o:before {

content: "\f1c7";

}

.fa-file-movie-o:before,

.fa-file-video-o:before {

content: "\f1c8";

}

.fa-file-code-o:before {

content: "\f1c9";

}

.fa-vine:before {

content: "\f1ca";

}

.fa-codepen:before {

content: "\f1cb";

}

.fa-jsfiddle:before {

content: "\f1cc";

}

.fa-life-bouy:before,

.fa-life-buoy:before,

.fa-life-saver:before,

.fa-support:before,

.fa-life-ring:before {

content: "\f1cd";

}

.fa-circle-o-notch:before {

content: "\f1ce";

}

.fa-ra:before,

.fa-resistance:before,

.fa-rebel:before {

content: "\f1d0";

}

.fa-ge:before,

.fa-empire:before {

content: "\f1d1";

}

.fa-git-square:before {

content: "\f1d2";

}

.fa-git:before {

content: "\f1d3";

}

.fa-y-combinator-square:before,

.fa-yc-square:before,

.fa-hacker-news:before {

content: "\f1d4";

}

.fa-tencent-weibo:before {

content: "\f1d5";

}

.fa-qq:before {

content: "\f1d6";

}

.fa-wechat:before,

.fa-weixin:before {

content: "\f1d7";

}

.fa-send:before,

.fa-paper-plane:before {

content: "\f1d8";

}

.fa-send-o:before,

.fa-paper-plane-o:before {

content: "\f1d9";

}

.fa-history:before {

content: "\f1da";

}

.fa-circle-thin:before {

content: "\f1db";

}

.fa-header:before {

content: "\f1dc";

}

.fa-paragraph:before {

content: "\f1dd";

}

.fa-sliders:before {

content: "\f1de";

}

.fa-share-alt:before {

content: "\f1e0";

}

.fa-share-alt-square:before {

content: "\f1e1";

}

.fa-bomb:before {

content: "\f1e2";

}

.fa-soccer-ball-o:before,

.fa-futbol-o:before {

content: "\f1e3";

}

.fa-tty:before {

content: "\f1e4";

}

.fa-binoculars:before {

content: "\f1e5";

}

.fa-plug:before {

content: "\f1e6";

}

.fa-slideshare:before {

content: "\f1e7";

}

.fa-twitch:before {

content: "\f1e8";

}

.fa-yelp:before {

content: "\f1e9";

}

.fa-newspaper-o:before {

content: "\f1ea";

}

.fa-wifi:before {

content: "\f1eb";

}

.fa-calculator:before {

content: "\f1ec";

}

.fa-paypal:before {

content: "\f1ed";

}

.fa-google-wallet:before {

content: "\f1ee";

}

.fa-cc-visa:before {

content: "\f1f0";

}

.fa-cc-mastercard:before {

content: "\f1f1";

}

.fa-cc-discover:before {

content: "\f1f2";

}

.fa-cc-amex:before {

content: "\f1f3";

}

.fa-cc-paypal:before {

content: "\f1f4";

}

.fa-cc-stripe:before {

content: "\f1f5";

}

.fa-bell-slash:before {

content: "\f1f6";

}

.fa-bell-slash-o:before {

content: "\f1f7";

}

.fa-trash:before {

content: "\f1f8";

}

.fa-copyright:before {

content: "\f1f9";

}

.fa-at:before {

content: "\f1fa";

}

.fa-eyedropper:before {

content: "\f1fb";

}

.fa-paint-brush:before {

content: "\f1fc";

}

.fa-birthday-cake:before {

content: "\f1fd";

}

.fa-area-chart:before {

content: "\f1fe";

}

.fa-pie-chart:before {

content: "\f200";

}

.fa-line-chart:before {

content: "\f201";

}

.fa-lastfm:before {

content: "\f202";

}

.fa-lastfm-square:before {

content: "\f203";

}

.fa-toggle-off:before {

content: "\f204";

}

.fa-toggle-on:before {

content: "\f205";

}

.fa-bicycle:before {

content: "\f206";

}

.fa-bus:before {

content: "\f207";

}

.fa-ioxhost:before {

content: "\f208";

}

.fa-angellist:before {

content: "\f209";

}

.fa-cc:before {

content: "\f20a";

}

.fa-shekel:before,

.fa-sheqel:before,

.fa-ils:before {

content: "\f20b";

}

.fa-meanpath:before {

content: "\f20c";

}

.fa-buysellads:before {

content: "\f20d";

}

.fa-connectdevelop:before {

content: "\f20e";

}

.fa-dashcube:before {

content: "\f210";

}

.fa-forumbee:before {

content: "\f211";

}

.fa-leanpub:before {

content: "\f212";

}

.fa-sellsy:before {

content: "\f213";

}

.fa-shirtsinbulk:before {

content: "\f214";

}

.fa-simplybuilt:before {

content: "\f215";

}

.fa-skyatlas:before {

content: "\f216";

}

.fa-cart-plus:before {

content: "\f217";

}

.fa-cart-arrow-down:before {

content: "\f218";

}

.fa-diamond:before {

content: "\f219";

}

.fa-ship:before {

content: "\f21a";

}

.fa-user-secret:before {

content: "\f21b";

}

.fa-motorcycle:before {

content: "\f21c";

}

.fa-street-view:before {

content: "\f21d";

}

.fa-heartbeat:before {

content: "\f21e";

}

.fa-venus:before {

content: "\f221";

}

.fa-mars:before {

content: "\f222";

}

.fa-mercury:before {

content: "\f223";

}

.fa-intersex:before,

.fa-transgender:before {

content: "\f224";

}

.fa-transgender-alt:before {

content: "\f225";

}

.fa-venus-double:before {

content: "\f226";

}

.fa-mars-double:before {

content: "\f227";

}

.fa-venus-mars:before {

content: "\f228";

}

.fa-mars-stroke:before {

content: "\f229";

}

.fa-mars-stroke-v:before {

content: "\f22a";

}

.fa-mars-stroke-h:before {

content: "\f22b";

}

.fa-neuter:before {

content: "\f22c";

}

.fa-genderless:before {

content: "\f22d";

}

.fa-facebook-official:before {

content: "\f230";

}

.fa-pinterest-p:before {

content: "\f231";

}

.fa-whatsapp:before {

content: "\f232";

}

.fa-server:before {

content: "\f233";

}

.fa-user-plus:before {

content: "\f234";

}

.fa-user-times:before {

content: "\f235";

}

.fa-hotel:before,

.fa-bed:before {

content: "\f236";

}

.fa-viacoin:before {

content: "\f237";

}

.fa-train:before {

content: "\f238";

}

.fa-subway:before {

content: "\f239";

}

.fa-medium:before {

content: "\f23a";

}

.fa-yc:before,

.fa-y-combinator:before {

content: "\f23b";

}

.fa-optin-monster:before {

content: "\f23c";

}

.fa-opencart:before {

content: "\f23d";

}

.fa-expeditedssl:before {

content: "\f23e";

}

.fa-battery-4:before,

.fa-battery:before,

.fa-battery-full:before {

content: "\f240";

}

.fa-battery-3:before,

.fa-battery-three-quarters:before {

content: "\f241";

}

.fa-battery-2:before,

.fa-battery-half:before {

content: "\f242";

}

.fa-battery-1:before,

.fa-battery-quarter:before {

content: "\f243";

}

.fa-battery-0:before,

.fa-battery-empty:before {

content: "\f244";

}

.fa-mouse-pointer:before {

content: "\f245";

}

.fa-i-cursor:before {

content: "\f246";

}

.fa-object-group:before {

content: "\f247";

}

.fa-object-ungroup:before {

content: "\f248";

}

.fa-sticky-note:before {

content: "\f249";

}

.fa-sticky-note-o:before {

content: "\f24a";

}

.fa-cc-jcb:before {

content: "\f24b";

}

.fa-cc-diners-club:before {

content: "\f24c";

}

.fa-clone:before {

content: "\f24d";

}

.fa-balance-scale:before {

content: "\f24e";

}

.fa-hourglass-o:before {

content: "\f250";

}

.fa-hourglass-1:before,

.fa-hourglass-start:before {

content: "\f251";

}

.fa-hourglass-2:before,

.fa-hourglass-half:before {

content: "\f252";

}

.fa-hourglass-3:before,

.fa-hourglass-end:before {

content: "\f253";

}

.fa-hourglass:before {

content: "\f254";

}

.fa-hand-grab-o:before,

.fa-hand-rock-o:before {

content: "\f255";

}

.fa-hand-stop-o:before,

.fa-hand-paper-o:before {

content: "\f256";

}

.fa-hand-scissors-o:before {

content: "\f257";

}

.fa-hand-lizard-o:before {

content: "\f258";

}

.fa-hand-spock-o:before {

content: "\f259";

}

.fa-hand-pointer-o:before {

content: "\f25a";

}

.fa-hand-peace-o:before {

content: "\f25b";

}

.fa-trademark:before {

content: "\f25c";

}

.fa-registered:before {

content: "\f25d";

}

.fa-creative-commons:before {

content: "\f25e";

}

.fa-gg:before {

content: "\f260";

}

.fa-gg-circle:before {

content: "\f261";

}

.fa-tripadvisor:before {

content: "\f262";

}

.fa-odnoklassniki:before {

content: "\f263";

}

.fa-odnoklassniki-square:before {

content: "\f264";

}

.fa-get-pocket:before {

content: "\f265";

}

.fa-wikipedia-w:before {

content: "\f266";

}

.fa-safari:before {

content: "\f267";

}

.fa-chrome:before {

content: "\f268";

}

.fa-firefox:before {

content: "\f269";

}

.fa-opera:before {

content: "\f26a";

}

.fa-internet-explorer:before {

content: "\f26b";

}

.fa-tv:before,

.fa-television:before {

content: "\f26c";

}

.fa-contao:before {

content: "\f26d";

}

.fa-500px:before {

content: "\f26e";

}

.fa-amazon:before {

content: "\f270";

}

.fa-calendar-plus-o:before {

content: "\f271";

}

.fa-calendar-minus-o:before {

content: "\f272";

}

.fa-calendar-times-o:before {

content: "\f273";

}

.fa-calendar-check-o:before {

content: "\f274";

}

.fa-industry:before {

content: "\f275";

}

.fa-map-pin:before {

content: "\f276";

}

.fa-map-signs:before {

content: "\f277";

}

.fa-map-o:before {

content: "\f278";

}

.fa-map:before {

content: "\f279";

}

.fa-commenting:before {

content: "\f27a";

}

.fa-commenting-o:before {

content: "\f27b";

}

.fa-houzz:before {

content: "\f27c";

}

.fa-vimeo:before {

content: "\f27d";

}

.fa-black-tie:before {

content: "\f27e";

}

.fa-fonticons:before {

content: "\f280";

}

.fa-reddit-alien:before {

content: "\f281";

}

.fa-edge:before {

content: "\f282";

}

.fa-credit-card-alt:before {

content: "\f283";

}

.fa-codiepie:before {

content: "\f284";

}

.fa-modx:before {

content: "\f285";

}

.fa-fort-awesome:before {

content: "\f286";

}

.fa-usb:before {

content: "\f287";

}

.fa-product-hunt:before {

content: "\f288";

}

.fa-mixcloud:before {

content: "\f289";

}

.fa-scribd:before {

content: "\f28a";

}

.fa-pause-circle:before {

content: "\f28b";

}

.fa-pause-circle-o:before {

content: "\f28c";

}

.fa-stop-circle:before {

content: "\f28d";

}

.fa-stop-circle-o:before {

content: "\f28e";

}

.fa-shopping-bag:before {

content: "\f290";

}

.fa-shopping-basket:before {

content: "\f291";

}

.fa-hashtag:before {

content: "\f292";

}

.fa-bluetooth:before {

content: "\f293";

}

.fa-bluetooth-b:before {

content: "\f294";

}

.fa-percent:before {

content: "\f295";

}

.fa-gitlab:before {

content: "\f296";

}

.fa-wpbeginner:before {

content: "\f297";

}

.fa-wpforms:before {

content: "\f298";

}

.fa-envira:before {

content: "\f299";

}

.fa-universal-access:before {

content: "\f29a";

}

.fa-wheelchair-alt:before {

content: "\f29b";

}

.fa-question-circle-o:before {

content: "\f29c";

}

.fa-blind:before {

content: "\f29d";

}

.fa-audio-description:before {

content: "\f29e";

}

.fa-volume-control-phone:before {

content: "\f2a0";

}

.fa-braille:before {

content: "\f2a1";

}

.fa-assistive-listening-systems:before {

content: "\f2a2";

}

.fa-asl-interpreting:before,

.fa-american-sign-language-interpreting:before {

content: "\f2a3";

}

.fa-deafness:before,

.fa-hard-of-hearing:before,

.fa-deaf:before {

content: "\f2a4";

}

.fa-glide:before {

content: "\f2a5";

}

.fa-glide-g:before {

content: "\f2a6";

}

.fa-signing:before,

.fa-sign-language:before {

content: "\f2a7";

}

.fa-low-vision:before {

content: "\f2a8";

}

.fa-viadeo:before {

content: "\f2a9";

}

.fa-viadeo-square:before {

content: "\f2aa";

}

.fa-snapchat:before {

content: "\f2ab";

}

.fa-snapchat-ghost:before {

content: "\f2ac";

}

.fa-snapchat-square:before {

content: "\f2ad";

}

.fa-pied-piper:before {

content: "\f2ae";

}

.fa-first-order:before {

content: "\f2b0";

}

.fa-yoast:before {

content: "\f2b1";

}

.fa-themeisle:before {

content: "\f2b2";

}

.fa-google-plus-circle:before,

.fa-google-plus-official:before {

content: "\f2b3";

}

.fa-fa:before,

.fa-font-awesome:before {

content: "\f2b4";

}

.fa-handshake-o:before {

content: "\f2b5";

}

.fa-envelope-open:before {

content: "\f2b6";

}

.fa-envelope-open-o:before {

content: "\f2b7";

}

.fa-linode:before {

content: "\f2b8";

}

.fa-address-book:before {

content: "\f2b9";

}

.fa-address-book-o:before {

content: "\f2ba";

}

.fa-vcard:before,

.fa-address-card:before {

content: "\f2bb";

}

.fa-vcard-o:before,

.fa-address-card-o:before {

content: "\f2bc";

}

.fa-user-circle:before {

content: "\f2bd";

}

.fa-user-circle-o:before {

content: "\f2be";

}

.fa-user-o:before {

content: "\f2c0";

}

.fa-id-badge:before {

content: "\f2c1";

}

.fa-drivers-license:before,

.fa-id-card:before {

content: "\f2c2";

}

.fa-drivers-license-o:before,

.fa-id-card-o:before {

content: "\f2c3";

}

.fa-quora:before {

content: "\f2c4";

}

.fa-free-code-camp:before {

content: "\f2c5";

}

.fa-telegram:before {

content: "\f2c6";

}

.fa-thermometer-4:before,

.fa-thermometer:before,

.fa-thermometer-full:before {

content: "\f2c7";

}

.fa-thermometer-3:before,

.fa-thermometer-three-quarters:before {

content: "\f2c8";

}

.fa-thermometer-2:before,

.fa-thermometer-half:before {

content: "\f2c9";

}

.fa-thermometer-1:before,

.fa-thermometer-quarter:before {

content: "\f2ca";

}

.fa-thermometer-0:before,

.fa-thermometer-empty:before {

content: "\f2cb";

}

.fa-shower:before {

content: "\f2cc";

}

.fa-bathtub:before,

.fa-s15:before,

.fa-bath:before {

content: "\f2cd";

}

.fa-podcast:before {

content: "\f2ce";

}

.fa-window-maximize:before {

content: "\f2d0";

}

.fa-window-minimize:before {

content: "\f2d1";

}

.fa-window-restore:before {

content: "\f2d2";

}

.fa-times-rectangle:before,

.fa-window-close:before {

content: "\f2d3";

}

.fa-times-rectangle-o:before,

.fa-window-close-o:before {

content: "\f2d4";

}

.fa-bandcamp:before {

content: "\f2d5";

}

.fa-grav:before {

content: "\f2d6";

}

.fa-etsy:before {

content: "\f2d7";

}

.fa-imdb:before {

content: "\f2d8";

}

.fa-ravelry:before {

content: "\f2d9";

}

.fa-eercast:before {

content: "\f2da";

}

.fa-microchip:before {

content: "\f2db";

}

.fa-snowflake-o:before {

content: "\f2dc";

}

.fa-superpowers:before {

content: "\f2dd";

}

.fa-wpexplorer:before {

content: "\f2de";

}

.fa-meetup:before {

content: "\f2e0";

}

.sr-only {

position: absolute;

width: 1px;

height: 1px;

padding: 0;

margin: -1px;

overflow: hidden;

clip: rect(0, 0, 0, 0);

border: 0;

}

.sr-only-focusable:active,

.sr-only-focusable:focus {

position: static;

width: auto;

height: auto;

margin: 0;

overflow: visible;

clip: auto;

}

const prefixCls = 'i-checkbox';

Component({

externalClasses: ,

relations: {

'../checkbox-group/index': {

type: 'parent'

}

},

properties: {

value: {

type: String,

value: ''

},

checked: {

type: Boolean,

value: false

},

disabled: {

type: Boolean,

value: false

},

color: {

type: String,

value: '#2d8cf0'

},

position: {

type: String,

value: 'left', //left right

observer: 'setPosition'

}

},

data: {

checked: true,

positionCls: `${prefixCls}-checkbox-left`,

},

attached() {

this.setPosition();

},

methods: {

changeCurrent(current) {

this.setData({ checked: current });

},

checkboxChange() {

if (this.data.disabled) return;

const item = { current: !this.data.checked, value: this.data.value };

const parent = this.getRelationNodes('../checkbox-group/index');

parent ? parent.emitEvent(item) : this.triggerEvent('change', item);

},

setPosition() {

this.setData({

positionCls: this.data.position.indexOf('left') !== -1 ? `${prefixCls}-checkbox-left` : `${prefixCls}-checkbox-right`,

});

}

}

});

<i-cell-group class="i-class">

<slot></slot>

</i-cell-group>

.i-col{float:left;box-sizing:border-box;width:0}.i-col-span-1{display:block;width:4.16666667%}.i-col-offset-1{margin-left:4.16666667%}.i-col-span-2{display:block;width:8.33333333%}.i-col-offset-2{margin-left:8.33333333%}.i-col-span-3{display:block;width:12.5%}.i-col-offset-3{margin-left:12.5%}.i-col-span-4{display:block;width:16.66666667%}.i-col-offset-4{margin-left:16.66666667%}.i-col-span-5{display:block;width:20.83333333%}.i-col-offset-5{margin-left:20.83333333%}.i-col-span-6{display:block;width:25%}.i-col-offset-6{margin-left:25%}.i-col-span-7{display:block;width:29.16666667%}.i-col-offset-7{margin-left:29.16666667%}.i-col-span-8{display:block;width:33.33333333%}.i-col-offset-8{margin-left:33.33333333%}.i-col-span-9{display:block;width:37.5%}.i-col-offset-9{margin-left:37.5%}.i-col-span-10{display:block;width:41.66666667%}.i-col-offset-10{margin-left:41.66666667%}.i-col-span-11{display:block;width:45.83333333%}.i-col-offset-11{margin-left:45.83333333%}.i-col-span-12{display:block;width:50%}.i-col-offset-12{margin-left:50%}.i-col-span-13{display:block;width:54.16666667%}.i-col-offset-13{margin-left:54.16666667%}.i-col-span-14{display:block;width:58.33333333%}.i-col-offset-14{margin-left:58.33333333%}.i-col-span-15{display:block;width:62.5%}.i-col-offset-15{margin-left:62.5%}.i-col-span-16{display:block;width:66.66666667%}.i-col-offset-16{margin-left:66.66666667%}.i-col-span-17{display:block;width:70.83333333%}.i-col-offset-17{margin-left:70.83333333%}.i-col-span-18{display:block;width:75%}.i-col-offset-18{margin-left:75%}.i-col-span-19{display:block;width:79.16666667%}.i-col-offset-19{margin-left:79.16666667%}.i-col-span-20{display:block;width:83.33333333%}.i-col-offset-20{margin-left:83.33333333%}.i-col-span-21{display:block;width:87.5%}.i-col-offset-21{margin-left:87.5%}.i-col-span-22{display:block;width:91.66666667%}.i-col-offset-22{margin-left:91.66666667%}.i-col-span-23{display:block;width:95.83333333%}.i-col-offset-23{margin-left:95.83333333%}.i-col-span-24{display:block;width:100%}.i-col-offset-24{margin-left:100%}

Component({

externalClasses: ,

relations: {

'../checkbox/index': {

type: 'child',

linked() {

this.changeCurrent();

},

linkChanged() {

this.changeCurrent();

},

unlinked() {

this.changeCurrent();

}

}

},

properties: {

current: {

type: Array,

value: ,

observer: 'changeCurrent'

},

},

methods: {

changeCurrent(val = this.data.current) {

let items = this.getRelationNodes('../checkbox/index');

const len = items.length;

if (len > 0) {

items.forEach(item => {

item.changeCurrent(val.indexOf(item.data.value) !== -1);

});

}

},

emitEvent(current) {

this.triggerEvent('change', current);

}

}

});

<view class="i-class i-col {{ span ? 'i-col-span-' + span : '' }} {{ offset ? 'i-col-offset-' + offset : '' }}"><slot></slot></view>

Component({

externalClasses: ,

relations: {

'../row/index': {

type: 'parent'

}

},

properties: {

span: {

value: 0,

type: Number

},

offset: {

value: 0,

type: Number

}

}

});

<view class="i-class i-collapse" bindcollapse="clickfn">

<slot></slot>

</view>

.i-collapse-item{padding:2px 8px;border-top:1px solid #dddee1}.i-collapse-item-title{vertical-align:middle}.i-collapse-item-title-wrap{padding:2px 0 0}.i-collapse-item-content{padding:6px;display:none}.i-collapse-item-show-content{display:block}.i-collapse-item-arrow{transition:transform .2s ease-in-out}.i-collapse-item-arrow-show{transition:transform .2s ease-in-out;transform:rotate(90deg)}

Component({

externalClasses: ,

relations: {

'../collapse-item/index': {

type: 'child'

}

},

properties: {

name: String,

accordion: Boolean

},

methods: {

clickfn(e) {

const params = e.detail;

const allList = this.getRelationNodes('../collapse-item/index');

allList.forEach((item) => {

if (params.name === item.data.name) {

item.setData({

showContent: 'i-collapse-item-show-content'

});

} else {

item.setData({

showContent: ''

});

}

});

},

}

});

<view id="{{name}}" class="i-class i-collapse-item ">

<view class="i-collapse-item-title-wrap" bindtap="trigger">

<i-icon size="16" type="enter" i-class="{{ showContent ? 'i-collapse-item-arrow-show' : 'i-collapse-item-arrow' }}"/>

<text class="i-collapse-item-title i-class-title">{{title}}</text>

</view>

<view class="i-collapse-item-content {{showContent}} i-class-content">

<slot name="content"></slot>

</view>

</view>

Component({

externalClasses: ,

relations: {

'../collapse/index': {

type: 'parent',

linked: function (target) {

const options = {

accordion: target.data.accordion

}

if (target.data.name === this.data.name) {

options.showContent = 'i-collapse-item-show-content';

}

this.setData(options);

}

}

},

properties: {

title: String,

name: String

},

data: {

showContent: '',

accordion: false

},

options: {

multipleSlots: true

},

methods: {

trigger(e) {

const data = this.data;

if (data.accordion) {

this.triggerEvent('collapse', {name: data.name}, {composed: true, bubbles: true});

} else {

this.setData({

showContent: data.showContent ? '' : 'i-collapse-item-show-content'

});

}

},

}

});

<text class="countdown-class">

{{time}}

</text>

.i-divider{width:100%;text-align:center;font-size:12px;position:relative;display:flex;align-items:center;justify-content:center}.i-divider-line{position:absolute;left:0;width:100%;height:1rpx;background-color:#f7f7f7;top:50%}.i-divider-content{background:#fff;position:relative;z-index:1;display:inline-block;padding:0 10px}

Component({

properties: {

target: Number,

showDay: Boolean,

callback: String,

format: Array,

clearTimer: Boolean

},

externalClasses: ,

data: {

time: '',

resultFormat: ,

changeFormat: false

},

ready() {

this.getFormat();

},

methods: {

getFormat() {

const data = this.data;

const len = data.format.length;

if (!data.showDay) data.resultFormat.push('');

if (len >= 3) {

for (let i = 0; i < len; i++) {

if (data.resultFormat.length >= 4) break;

if (data.format) {

data.resultFormat.push(data.format.toString());

}

}

if (data.resultFormat.length >= 4) data.changeFormat = true;

}

this.getLastTime();

},

init() {

const self = this;

setTimeout(function () {

self.getLastTime.call(self);

}, 1000);

},

getLastTime() {

const data = this.data;

const gapTime = Math.ceil((data.target - new Date().getTime()) / 1000);

let result = '';

let time = '00:00:00';

let day = '00';

const format = data.resultFormat;

if (gapTime > 0) {

day = this.formatNum(parseInt(gapTime / 86400));

let lastTime = gapTime % 86400;

const hour = this.formatNum(parseInt(lastTime / 3600));

lastTime = lastTime % 3600;

const minute = this.formatNum(parseInt(lastTime / 60));

const second = this.formatNum(lastTime % 60);

if (data.changeFormat) time = `${hour}${format}${minute}${format}${second}${format}`;

else time = `${hour}:${minute}:${second}`;

if (!data.clearTimer) this.init.call(this);

} else {

this.endfn();

}

if (data.showDay) {

if (data.changeFormat) {

result = `${day}${format} ${time}`;

} else {

result = `${day}d ${time}`;

}

} else {

result = time;

}

this.setData({

time: result

});

},

formatNum(num) {

return num > 9 ? num : `0${num}`;

},

endfn() {

this.triggerEvent('callback', {});

}

}

});

<view class="i-divider i-class" style="{{parse.getStyle(color,size,height)}}">

<view class="i-divider-content" wx:if="{{content !== ''}}">

{{content}}

</view>

<view class="i-divider-content" wx:else>

<slot></slot>

</view>

<view class="i-divider-line" style="background:{{lineColor}}"></view>

</view>

<wxs module="parse">

module.exports = {

getStyle : function(color,size,height){

var color = 'color:' + color +';';

var size = 'font-size:' + size + 'px;';

var height = 'height:' + height+'px;'

return color + size + height;

}

}

</wxs>

.i-drawer{visibility:hidden}.i-drawer-show{visibility:visible}.i-drawer-show .i-drawer-mask{display:block;opacity:1}.i-drawer-show .i-drawer-container{opacity:1}.i-drawer-show.i-drawer-left .i-drawer-container,.i-drawer-show.i-drawer-right .i-drawer-container{transform:translate3d(0,-50%,0)}.i-drawer-mask{opacity:0;position:fixed;top:0;left:0;right:0;bottom:0;z-index:6;background:rgba(0,0,0,.6);transition:all .3s ease-in-out}.i-drawer-container{position:fixed;left:50%;top:50%;transform:translate3d(-50%,-50%,0);transform-origin:center;transition:all .3s ease-in-out;z-index:7;opacity:0}.i-drawer-left .i-drawer-container{left:0;top:50%;transform:translate3d(-100%,-50%,0)}.i-drawer-right .i-drawer-container{right:0;top:50%;left:auto;transform:translate3d(100%,-50%,0)}

Component({

externalClasses: ,

properties: {

content: {

type: String,

value: ''

},

height : {

type: Number,

value: 48

},

color : {

type : String,

value : '#80848f'

},

lineColor : {

type : String,

value : '#e9eaec'

},

size : {

type: String,

value: 12

}

}

});

<view class="i-class i-drawer {{ visible ? 'i-drawer-show' : '' }} {{ 'i-drawer-' + mode }}">

<view wx:if="{{ mask }}" class="i-drawer-mask" bindtap="handleMaskClick"></view>

<view class="i-drawer-container">

<slot></slot>

</view>

</view>

.i-grid{border-top:1rpx solid #e9eaec;border-left:1rpx solid #e9eaec;overflow:hidden}

Component({

externalClasses: ,

properties: {

visible: {

type: Boolean,

value: false

},

mask: {

type: Boolean,

value: true

},

maskClosable: {

type: Boolean,

value: true

},

mode: {

type: String,

value: 'left' // left right

}

},

data: {},

methods: {

handleMaskClick() {

if (!this.data.maskClosable) {

return;

}

this.triggerEvent('close', {});

}

}

});

<view class="i-class i-grid"><slot></slot></view>.i-grid-icon{display:block;width:28px;height:28px;margin:0 auto}.i-grid-icon image{width:100%;height:100%}

Component({

externalClasses: ,

relations: {

'../grid-item/index': {

type: 'child',

linked () {

this.setGridItemWidth();

},

linkChanged () {

this.setGridItemWidth();

},

unlinked () {

this.setGridItemWidth();

}

}

},

methods: {

setGridItemWidth () {

const nodes = this.getRelationNodes('../grid-item/index');

// const len = nodes.length;

// if (len < 3) {

// nodes.forEach(item => {

// item.setData({

// 'width': '33.33%'

// });

// });

// } else {

// const width = 100 / nodes.length;

// nodes.forEach(item => {

// item.setData({

// 'width': width + '%'

// });

// });

// }

const width = 100 / nodes.length;

nodes.forEach(item => {

item.setData({

'width': width + '%'

});

});

}

},

ready () {

this.setGridItemWidth();

}

});

<view class="i-class i-grid-icon"><slot></slot></view>.i-grid-item{position:relative;float:left;padding:20px 10px;width:33.33333333%;box-sizing:border-box;border-right:1rpx solid #e9eaec;border-bottom:1rpx solid #e9eaec}

Component({

externalClasses: ,

relations: {

'../grid-item/index': {

type: 'parent'

}

},

});

<view class="i-class i-grid-item" style="width: {{ width }}"><slot></slot></view>.i-grid-label{margin-top:5px;display:block;text-align:center;color:#1c2438;font-size:14px;white-space:nowrap;text-overflow:ellipsis;overflow:hidden}

Component({

externalClasses: ,

options: {

multipleSlots: true

},

properties: {

//info, success, warning, error

type: {

type: String,

value: 'info'

},

closable: {

type: Boolean,

value: false

},

showIcon: {

type: Boolean,

default: false

},

desc: {

type: Boolean,

default: false

},

},

data: {

closed: false

},

methods: {

handleTap() {

this.setData({

closed: !this.data.closed,

});

this.triggerEvent('close');

},

}

});

<view class="i-class i-alert {{'i-alert-'+type}} {{showIcon?'i-alert-with-icon':''}} {{desc?'i-alert-with-desc':''}}" wx:if="{{!closed}}">

<view wx:if="{{ showIcon }}" class="i-alert-icon">

<i-icon type="prompt" wx:if="{{ type === 'info' }}" size="{{desc?24:16}}"></i-icon>

<i-icon type="success" wx:if="{{ type === 'success' }}" size="{{desc?24:16}}"></i-icon>

<i-icon type="warning" wx:if="{{ type === 'warning' }}" size="{{desc?24:16}}"></i-icon>

<i-icon type="delete" wx:if="{{ type === 'error' }}" size="{{desc?24:16}}"></i-icon>

</view>

<slot></slot>

<view class="i-alert-desc">

<slot name="desc"></slot>

</view>

<view class="i-alert-close" wx:if="{{ closable }}" bindtap="handleTap">

<i-icon type="close"></i-icon>

</view>

</view>

.i-alert{position:relative;margin:10px;padding:8px 48px 8px 16px;font-size:14px;border-radius:2px;color:#fff;background:#f7f7f7;color:#495060}.i-alert.i-alert-with-icon{padding:8px 48px 8px 38px}.i-alert-info{color:#fff;background:#2db7f5}.i-alert-success{color:#fff;background:#19be6b}.i-alert-warning{color:#fff;background:#f90}.i-alert-error{color:#fff;background:#ed3f14}.i-alert-icon{position:absolute;top:9px;left:16px;font-size:14px}.i-alert-desc{font-size:12px}.i-alert-with-desc{padding:16px;position:relative}.i-alert-with-desc.i-alert-with-icon{padding:16px 16px 16px 69px}.i-alert-with-desc .i-alert-icon{top:50%;left:24px;margin-top:-21px;font-size:28px}.i-alert-close{font-size:12px;position:absolute;right:16px;top:8px;overflow:hidden;cursor:pointer}

Component({

externalClasses: ,

relations: {

'../grid/index': {

type: 'parent'

},

'../grid-icon/index': {

type: 'child'

}

},

data: {

width: '33.33%'

}

});

<view class="i-class i-grid-label"><slot></slot></view>@font-face{font-family:iconfont;src:url('data:application/x-font-woff;charset=utf-8;base64,') format('woff')}.i-icon{display:inline-block;font-family:iconfont;speak:none;font-style:normal;font-weight:400;font-variant:normal;text-transform:none;text-rendering:auto;line-height:1;-webkit-font-smoothing:antialiased;-moz-osx-font-smoothing:grayscale;vertical-align:middle}.i-icon-accessory:before{content:"\e6dd"}.i-icon-activity:before{content:"\e6de"}.i-icon-activity\_fill:before{content:"\e6df"}.i-icon-add:before{content:"\e6e0"}.i-icon-addressbook\_fill:before{content:"\e6e2"}.i-icon-addressbook:before{content:"\e6e3"}.i-icon-barrage\_fill:before{content:"\e6e4"}.i-icon-barrage:before{content:"\e6e5"}.i-icon-browse\_fill:before{content:"\e6e6"}.i-icon-browse:before{content:"\e6e7"}.i-icon-brush:before{content:"\e6e8"}.i-icon-brush\_fill:before{content:"\e6e9"}.i-icon-businesscard\_fill:before{content:"\e6ea"}.i-icon-businesscard:before{content:"\e6eb"}.i-icon-camera\_fill:before{content:"\e6ec"}.i-icon-camera:before{content:"\e6ed"}.i-icon-clock\_fill:before{content:"\e6ee"}.i-icon-clock:before{content:"\e6ef"}.i-icon-close:before{content:"\e6f0"}.i-icon-collection\_fill:before{content:"\e6f1"}.i-icon-collection:before{content:"\e6f2"}.i-icon-computer\_fill:before{content:"\e6f3"}.i-icon-computer:before{content:"\e6f4"}.i-icon-coordinates\_fill:before{content:"\e6f5"}.i-icon-coordinates:before{content:"\e6f6"}.i-icon-coupons\_fill:before{content:"\e6f7"}.i-icon-coupons:before{content:"\e6f8"}.i-icon-createtask\_fill:before{content:"\e6f9"}.i-icon-createtask:before{content:"\e6fa"}.i-icon-customerservice\_fill:before{content:"\e6fb"}.i-icon-customerservice:before{content:"\e6fc"}.i-icon-delete\_fill:before{content:"\e6fd"}.i-icon-delete:before{content:"\e6fe"}.i-icon-document:before{content:"\e6ff"}.i-icon-document\_fill:before{content:"\e700"}.i-icon-dynamic\_fill:before{content:"\e701"}.i-icon-dynamic:before{content:"\e702"}.i-icon-editor:before{content:"\e703"}.i-icon-eit:before{content:"\e704"}.i-icon-emoji\_fill:before{content:"\e705"}.i-icon-emoji:before{content:"\e706"}.i-icon-enter:before{content:"\e707"}.i-icon-enterinto:before{content:"\e708"}.i-icon-enterinto\_fill:before{content:"\e709"}.i-icon-feedback\_fill:before{content:"\e70a"}.i-icon-feedback:before{content:"\e70b"}.i-icon-flag\_fill:before{content:"\e70c"}.i-icon-flag:before{content:"\e70d"}.i-icon-flashlight:before{content:"\e70e"}.i-icon-flashlight\_fill:before{content:"\e70f"}.i-icon-fullscreen:before{content:"\e710"}.i-icon-group:before{content:"\e711"}.i-icon-group\_fill:before{content:"\e712"}.i-icon-homepage\_fill:before{content:"\e713"}.i-icon-homepage:before{content:"\e714"}.i-icon-integral\_fill:before{content:"\e715"}.i-icon-integral:before{content:"\e716"}.i-icon-interactive\_fill:before{content:"\e717"}.i-icon-interactive:before{content:"\e718"}.i-icon-keyboard:before{content:"\e719"}.i-icon-label:before{content:"\e71a"}.i-icon-label\_fill:before{content:"\e71b"}.i-icon-like\_fill:before{content:"\e71c"}.i-icon-like:before{content:"\e71d"}.i-icon-live\_fill:before{content:"\e71e"}.i-icon-live:before{content:"\e71f"}.i-icon-lock\_fill:before{content:"\e720"}.i-icon-lock:before{content:"\e721"}.i-icon-mail:before{content:"\e722"}.i-icon-mail\_fill:before{content:"\e723"}.i-icon-message:before{content:"\e724"}.i-icon-message\_fill:before{content:"\e725"}.i-icon-mine:before{content:"\e726"}.i-icon-mine\_fill:before{content:"\e727"}.i-icon-mobilephone\_fill:before{content:"\e728"}.i-icon-mobilephone:before{content:"\e729"}.i-icon-more:before{content:"\e72a"}.i-icon-narrow:before{content:"\e72b"}.i-icon-offline\_fill:before{content:"\e72c"}.i-icon-offline:before{content:"\e72d"}.i-icon-other:before{content:"\e72e"}.i-icon-picture\_fill:before{content:"\e72f"}.i-icon-picture:before{content:"\e730"}.i-icon-play:before{content:"\e731"}.i-icon-play\_fill:before{content:"\e732"}.i-icon-playon\_fill:before{content:"\e733"}.i-icon-playon:before{content:"\e734"}.i-icon-praise\_fill:before{content:"\e735"}.i-icon-praise:before{content:"\e736"}.i-icon-prompt\_fill:before{content:"\e737"}.i-icon-prompt:before{content:"\e738"}.i-icon-redpacket\_fill:before{content:"\e739"}.i-icon-redpacket:before{content:"\e73a"}.i-icon-refresh:before{content:"\e73b"}.i-icon-remind\_fill:before{content:"\e73c"}.i-icon-remind:before{content:"\e73d"}.i-icon-return:before{content:"\e73e"}.i-icon-right:before{content:"\e73f"}.i-icon-scan:before{content:"\e740"}.i-icon-send:before{content:"\e741"}.i-icon-service\_fill:before{content:"\e742"}.i-icon-service:before{content:"\e743"}.i-icon-setup\_fill:before{content:"\e744"}.i-icon-setup:before{content:"\e745"}.i-icon-share\_fill:before{content:"\e746"}.i-icon-share:before{content:"\e747"}.i-icon-success\_fill:before{content:"\e748"}.i-icon-success:before{content:"\e749"}.i-icon-suspend:before{content:"\e74a"}.i-icon-switch:before{content:"\e74b"}.i-icon-systemprompt\_fill:before{content:"\e74c"}.i-icon-systemprompt:before{content:"\e74d"}.i-icon-tailor:before{content:"\e74e"}.i-icon-task:before{content:"\e74f"}.i-icon-task\_fill:before{content:"\e750"}.i-icon-tasklist\_fill:before{content:"\e751"}.i-icon-tasklist:before{content:"\e752"}.i-icon-time\_fill:before{content:"\e753"}.i-icon-time:before{content:"\e754"}.i-icon-translation\_fill:before{content:"\e755"}.i-icon-translation:before{content:"\e756"}.i-icon-trash:before{content:"\e757"}.i-icon-trash\_fill:before{content:"\e758"}.i-icon-undo:before{content:"\e759"}.i-icon-video:before{content:"\e75a"}.i-icon-video\_fill:before{content:"\e75b"}.i-icon-warning\_fill:before{content:"\e75c"}.i-icon-warning:before{content:"\e75d"}.i-icon-search:before{content:"\e75e"}.i-icon-searchfill:before{content:"\e75f"}.i-icon-publishgoods\_fill:before{content:"\e760"}.i-icon-shop\_fill:before{content:"\e761"}.i-icon-transaction\_fill:before{content:"\e762"}.i-icon-packup:before{content:"\e763"}.i-icon-unfold:before{content:"\e764"}.i-icon-financial\_fill:before{content:"\e765"}.i-icon-commodity:before{content:"\e766"}

Component({

externalClasses: ,

relations: {

'../grid-item/index': {

type: 'parent'

}

},

});

<view class="i-class i-icon {{ type === '' ? '' : 'i-icon-' + type }} {{ custom }}" style="font-size: {{ size }}px; {{ color ? 'color:' + color : '' }}"></view>.i-index{width:100%;height:100%}.i-index-line{position:absolute;left:0;width:100%;height:1rpx;background-color:#f7f7f7;top:50%}.i-index-content{background:#fff;position:relative;z-index:1;display:inline-block;padding:0 10px}.i-index-fixed{position:fixed;right:0;top:50%;z-index:10;padding-left:10px;transform:translateY(-50%)}.i-index-fixed-item{display:block;height:18px;line-height:18px;padding:0 5px;text-align:center;color:#2d8cf0;font-size:12px;border-radius:50%}.i-index-fixed-item-current{background:#2d8cf0;color:#fff}.i-index-tooltip{position:fixed;left:50%;top:50%;transform:translate3d(-50%,-50%,0);background:rgba(0,0,0,.7);color:#fff;font-size:24px;border-radius:50%;width:80px;height:80px;line-height:80px;text-align:center}

Component({

externalClasses: ,

properties: {

type: {

type: String,

value: ''

},

custom: {

type: String,

value: ''

},

size: {

type: Number,

value: 14

},

color: {

type: String,

value: ''

}

}

});

<view class="i-index i-class">

<scroll-view

scroll-y

style="{{parse.setScrollStyle(height)}}"

bindscroll="handlerScroll"

scroll-top="{{scrollTop}}">

<slot></slot>

<view class="i-index-fixed"

catchtouchstart="handlerTouchMove"

catchtouchmove="handlerTouchMove"

catchtouchend="handlerTouchEnd">

<view class="i-index-fixed-item"

wx:for="{{fixedData}}"

wx:key="{{index}}"

data-index="{{index}}"

catchtap="handlerFixedTap">

{{item}}

</view>

</view>

<view class="i-index-tooltip" style="{{ isTouches ? 'display:block' : 'display:none' }}">{{currentName}}</view>

</scroll-view>

</view>

<wxs module="parse">

module.exports = {

setScrollStyle : function(height){

var units = ;

var hasUnits = false;

for( var i = 0; i < units.length;i++ ){

var u = units;

if( height.indexOf( u ) > -1 ){

hasUnits = true;

break;

}

}

return 'height:'+ ( hasUnits ? height : height+'px' );

}

}

</wxs>

.i-index-item-header{height:30px;line-height:30px;background:#eee;font-size:14px;padding-left:10px;width:100%;box-sizing:border-box}.i-index-item-content{font-size:14px}

Component({

externalClasses: ,

properties : {

height : {

type : String,

value : '300'

},

itemHeight : {

type : Number,

value : 18

}

},

relations : {

'../index-item/index' : {

type : 'child',

linked(){

this.\_updateDataChange();

},

linkChanged () {

this.\_updateDataChange();

},

unlinked () {

this.\_updateDataChange();

}

}

},

data : {

scrollTop : 0,

fixedData : ,

current : 0,

timer : null,

startTop : 0,

itemLength : 0,

currentName : '',

isTouches : false

},

methods : {

loop(){},

\_updateDataChange( ){

const indexItems = this.getRelationNodes('../index-item/index');

const len = indexItems.length;

const fixedData = this.data.fixedData;

/\*

\* 使用函数节流限制重复去设置数组内容进而限制多次重复渲染

\* 暂时没有研究微信在渲染的时候是否会进行函数节流

\*/

if (len > 0) {

if( this.data.timer ){

clearTimeout( this.data.timer )

this.setData({

timer : null

})

}

this.data.timer = setTimeout(()=>{

const data = ;

indexItems.forEach((item) => {

if( item.data.name && fixedData.indexOf( item.data.name ) === -1 ){

data.push(item.data.name);

item.updateDataChange();

}

})

this.setData({

fixedData : data,

itemLength : indexItems.length

})

//组件加载完成之后重新设置顶部高度

this.setTouchStartVal();

},0);

this.setData({

timer : this.data.timer

})

}

},

handlerScroll(event){

const detail = event.detail;

const scrollTop = detail.scrollTop;

const indexItems = this.getRelationNodes('../index-item/index');

indexItems.forEach((item,index)=>{

let data = item.data;

let offset = data.top + data.height;

if( scrollTop < offset && scrollTop >= data.top ){

this.setData({

current : index,

currentName : data.currentName

})

}

})

},

getCurrentItem(index){

const indexItems = this.getRelationNodes('../index-item/index');

let result = {};

result = indexItems.data;

result.total = indexItems.length;

return result;

},

triggerCallback(options){

this.triggerEvent('change',options)

},

handlerFixedTap(event){

const eindex = event.currentTarget.dataset.index;

const item = this.getCurrentItem(eindex);

this.setData({

scrollTop : item.top,

currentName : item.currentName,

isTouches : true

})

this.triggerCallback({

index : eindex,

current : item.currentName

})

},

handlerTouchMove(event){

const data = this.data;

const touches = event.touches || {};

const pageY = touches.pageY;

const rest = pageY - data.startTop;

let index = Math.ceil( rest/data.itemHeight );

index = index >= data.itemLength ? data.itemLength -1 : index;

const movePosition = this.getCurrentItem(index);

/\*

\* 当touch选中的元素和当前currentName不相等的时候才震动一下

\* 微信震动事件

\*/

if( movePosition.name !== this.data.currentName ){

wx.vibrateShort();

}

this.setData({

scrollTop : movePosition.top,

currentName : movePosition.name,

isTouches : true

})

this.triggerCallback({

index : index,

current : movePosition.name

})

},

handlerTouchEnd(){

this.setData({

isTouches : false

})

},

setTouchStartVal(){

const className = '.i-index-fixed';

const query = wx.createSelectorQuery().in(this);

query.select( className ).boundingClientRect((res)=>{

this.setData({

startTop : res.top

})

}).exec()

}

}

})<view class="i-index-item i-class">

<view class="i-index-item-header">{{name}}</view>

<view class="i-index-item-content">

<slot></slot>

</view>

</view>

<wxs module="parse">

module.exports = {

}

</wxs>

.i-cell{position:relative;padding:12px 15px;display:flex;background:#fff;align-items:center;line-height:1.4;font-size:14px;overflow:hidden}.i-cell::after{content:'';position:absolute;top:0;left:0;width:200%;height:200%;transform:scale(.5);transform-origin:0 0;pointer-events:none;box-sizing:border-box;border:0 solid #e9eaec;border-bottom-width:1px;left:15px;right:0}.i-cell-last::after{display:none}.i-cell-icon{margin-right:5px}.i-cell-icon:empty{display:none}.i-cell-bd{flex:1}.i-cell-text{line-height:24px;font-size:14px}.i-cell-desc{line-height:1.2;font-size:12px;color:#80848f}.i-cell-ft{position:relative;text-align:right;color:#495060}.i-cell-access .i-cell-ft{padding-right:13px}.i-cell-access .i-cell-ft::after{content:" ";display:inline-block;width:6px;height:6px;position:absolute;top:50%;right:2px;border-width:2px 2px 0 0;border-color:#dddee1;border-style:solid;transform:translateY(-50%) matrix(.71,.71,-.71,.71,0,0)}.i-input{padding:7px 15px;color:#495060}.i-input-wrapped{margin:10px 15px;background-color:#fff}.i-input-wrapped::after{left:0;border-width:1px;border-radius:4px}.i-input-error{color:#ed3f14}.i-input-title{color:#495060;min-width:65px;padding-right:10px}.i-input-input{flex:1;line-height:1.6;padding:4px 0;min-height:22px;height:auto;font-size:14px}.i-input-placeholder{font-size:14px}.i-input-input-right{text-align:right}.i-input.i-input-wrapped::after{display:block}.i-input-wrapped.i-input-error::after{border-color:#ed3f14}

Component({

externalClasses: ,

properties : {

name : {

type : String,

value : ''

}

},

relations : {

'../index/index' : {

type : 'parent'

}

},

data : {

top : 0,

height : 0,

currentName : ''

},

methods: {

updateDataChange() {

const className = '.i-index-item';

const query = wx.createSelectorQuery().in(this);

query.select( className ).boundingClientRect((res)=>{

this.setData({

top : res.top,

height : res.height,

currentName : this.data.name

})

}).exec()

}

}

})<view class="i-class i-cell i-input {{ error ? 'i-input-error' : '' }} {{ mode === 'wrapped' ? 'i-input-wrapped' : '' }}">

<view wx:if="{{ title }}" class="i-cell-hd i-input-title">{{ title }}</view>

<textarea

wx:if="{{ type === 'textarea' }}"

auto-height

disabled="{{ disabled }}"

focus="{{ autofocus }}"

value="{{ value }}"

placeholder="{{ placeholder }}"

maxlength="{{ maxlength }}"

class="i-input-input i-cell-bd {{ right ? 'i-input-input-right' : '' }}"

placeholder-class="i-input-placeholder"

bindinput="handleInputChange"

bindfocus="handleInputFocus"

bindblur="handleInputBlur"

></textarea>

<input

wx:else

type="{{ type }}"

disabled="{{ disabled }}"

focus="{{ autofocus }}"

value="{{ value }}"

placeholder="{{ placeholder }}"

maxlength="{{ maxlength }}"

class="i-input-input i-cell-bd {{ right ? 'i-input-input-right' : '' }}"

placeholder-class="i-input-placeholder"

bindinput="handleInputChange"

bindfocus="handleInputFocus"

bindblur="handleInputBlur"

/>

</view>

.i-input-number{color:#495060}.i-input-number view{display:inline-block;line-height:20px;padding:5px 0;text-align:center;min-width:40px;box-sizing:border-box;vertical-align:middle;font-size:12px;border:1rpx solid #dddee1}.i-input-number-minus{border-right:none;border-radius:2px 0 0 2px}.i-input-number-plus{border-left:none;border-radius:0 2px 2px 0}.i-input-number-text{border:1rpx solid #dddee1;display:inline-block;text-align:center;vertical-align:middle;height:30px;width:40px;min-height:auto;font-size:12px;line-height:30px}.i-input-number-disabled{border-color:#dddee1;color:#bbbec4;background:#f7f7f7}

Component({

behaviors: ,

externalClasses: ,

properties: {

title: {

type: String

},

// text || textarea || password || number

type: {

type: String,

value: 'text'

},

disabled: {

type: Boolean,

value: false

},

placeholder: {

type: String,

value: ''

},

autofocus: {

type: Boolean,

value: false

},

mode: {

type: String,

value: 'normal'

},

right: {

type: Boolean,

value: false

},

error: {

type: Boolean,

value: false

},

maxlength: {

type: Number

}

},

methods: {

handleInputChange(event) {

const { detail = {} } = event;

const { value = '' } = detail;

this.setData({ value });

this.triggerEvent('change', event);

},

handleInputFocus(event) {

this.triggerEvent('focus', event);

},

handleInputBlur(event) {

this.triggerEvent('blur', event);

}

}

});

<view class="i-class i-input-number i-input-number-size-{{ size }}">

<view class="i-input-number-minus {{ value <= min ? 'i-input-number-disabled' : '' }}" data-disabled="{{ value <= min }}" bindtap="handleMinus">-</view>

<input class="i-input-number-text {{ min >= max ? 'i-input-number-disabled' : '' }}" type="number" value="{{ value }}" disabled="{{ min >= max }}" bindblur="handleBlur" />

<view class="i-input-number-plus {{ value >= max ? 'i-input-number-disabled' : '' }}" data-disabled="{{ value >= max }}" bindtap="handlePlus">+</view>

</view>

.i-load-more{width:65%;margin:1.5em auto;line-height:1.6em;font-size:14px;text-align:center}.i-load-more-loading{display:inline-block;margin-right:12px;vertical-align:middle;width:14px;height:14px;background:0 0;border-radius:50%;border:2px solid #e9eaec;border-color:#e9eaec #e9eaec #e9eaec #2d8cf0;animation:btn-spin .6s linear;animation-iteration-count:infinite}.i-load-more-tip{display:inline-block;vertical-align:middle;color:#495060}.i-load-more-line{border-top:1px solid #dddee1;display:flex;border-top:0}.i-load-more-line::before{position:relative;top:-1px;-webkit-box-flex:1;-webkit-flex:1;flex:1;content:'';border-top:1px solid #dddee1}.i-load-more-line::after{position:relative;top:-1px;-webkit-box-flex:1;-webkit-flex:1;flex:1;content:'';border-top:1px solid #dddee1}.i-load-more-line .i-load-more-tip{position:relative;top:-.9em;padding:0 .55em}.i-load-more-empty{width:4px;height:4px;border-radius:50%;background-color:#e5e5e5;display:inline-block;position:relative;vertical-align:0;top:-.16em}@keyframes btn-spin{0%{transform:rotate(0)}100%{transform:rotate(360deg)}}

function addNum (num1, num2) {

let sq1, sq2, m;

try {

sq1 = num1.toString().split('.').length;

}

catch (e) {

sq1 = 0;

}

try {

sq2 = num2.toString().split('.').length;

}

catch (e) {

sq2 = 0;

}

m = Math.pow(10, Math.max(sq1, sq2));

return (Math.round(num1 \* m) + Math.round(num2 \* m)) / m;

}

Component({

externalClasses: ,

properties: {

// small || default || large

size: String,

value: {

type: Number,

value: 1

},

min: {

type: Number,

value: -Infinity

},

max: {

type: Number,

value: Infinity

},

step: {

type: Number,

value: 1

}

},

methods: {

handleChangeStep(e, type) {

const { dataset = {} } = e.currentTarget;

const { disabled } = dataset;

const { step } = this.data;

let { value } = this.data;

if (disabled) return null;

if (type === 'minus') {

value = addNum(value, -step);

} else if (type === 'plus') {

value = addNum(value, step);

}

if (value < this.data.min || value > this.data.max) return null;

this.handleEmit(value, type);

},

handleMinus(e) {

this.handleChangeStep(e, 'minus');

},

handlePlus(e) {

this.handleChangeStep(e, 'plus');

},

handleBlur(e) {

let { value } = e.detail;

const { min, max } = this.data;

if (!value) {

setTimeout(() => {

this.handleEmit(value);

}, 16);

return;

}

value = +value;

if (value > max) {

value = max;

} else if (value < min) {

value = min;

}

this.handleEmit(value);

},

handleEmit (value, type) {

const data = {

value: value

};

if (type) data.type = type;

this.triggerEvent('change', data);

}

}

});

<view class="i-class i-load-more {{ loading ? '' : 'i-load-more-line' }}">

<view class="i-load-more-loading" wx:if="{{ loading }}"></view>

<view class="i-load-more-tip">

<view wx:if="{{ tip !== '' }}">{{ tip }}</view>

<view wx:elif="{{ tip === '' && loading }}">正在加载</view>

<view class="i-load-more-empty" wx:else></view>

</view>

</view>.i-message{display:block;width:100%;min-height:32px;line-height:2.3;position:fixed;top:0;left:0;right:0;background:#2d8cf0;color:#fff;text-align:center;font-size:14px;z-index:1010;opacity:0;-webkit-transform:translateZ(0) translateY(-100%);transition:all .4s ease-in-out}.i-message-show{-webkit-transform:translateZ(0) translateY(0);opacity:1}.i-message-default{background:#2d8cf0}.i-message-success{background:#19be6b}.i-message-warning{background:#f90}.i-message-error{background:#ed3f14}

Component({

externalClasses: ,

properties: {

loading: {

type: Boolean,

value: true

},

tip: {

type: String,

value: ''

}

},

});

<view class="i-class i-message i-message-{{type}} {{ visible ? 'i-message-show' : '' }}">

{{ content }}

</view>.i-modal{position:fixed;overflow:auto;top:0;right:0;bottom:0;left:0;height:100%;z-index:1000;display:flex;outline:0;-webkit-box-align:center;align-items:center;-webkit-box-pack:center;justify-content:center;transform:translateZ(1px);opacity:0;visibility:hidden}.i-modal-show{visibility:visible;opacity:1}.i-modal-mask{position:fixed;top:0;left:0;right:0;bottom:0;background:rgba(0,0,0,.7);z-index:1000;transition:all .2s ease-in-out;opacity:0;visibility:hidden}.i-modal-mask-show{opacity:1;visibility:visible}.i-modal-main{width:270px;position:relative}.i-modal-content{border-radius:7px;padding-top:15px;position:relative;background-color:#fff;border:0;background-clip:padding-box;text-align:center;height:100%;overflow:hidden}.i-modal-body{max-height:100px;margin-bottom:15px;font-size:14px;color:#80848f;height:100%;line-height:1.5;overflow:auto}.i-modal-title{padding:6px 15px 15px;margin:0;font-size:18px;line-height:1;color:#1c2438;text-align:center}.i-modal-actions{margin:0 1px}.i-modal-action-vertical{position:relative}.i-modal-action-vertical:after{content:'';position:absolute;top:0;left:0;width:200%;height:200%;transform:scale(.5);transform-origin:0 0;pointer-events:none;box-sizing:border-box;border:0 solid #e9eaec;border-top-width:1px}.i-modal-grid{border-radius:0 0 7px 7px;border-left:none}.i-modal-grid-item,.i-modal-grid-item-last{padding:0;border-bottom:none}.i-modal-grid-item-last{border-right:none}.i-modal-btn-ok{color:#2d8cf0!important}.i-modal-btn-loading{display:inline-block;vertical-align:middle;margin-right:10px;width:12px;height:12px;background:0 0;border-radius:50%;border:2px solid #e5e5e5;border-color:#666 #e5e5e5 #e5e5e5 #e5e5e5;animation:btn-spin .6s linear;animation-iteration-count:infinite}.i-modal-btn-text{display:inline-block;vertical-align:middle}.i-modal-btn-icon{font-size:14px!important;margin-right:4px}@keyframes btn-spin{0%{transform:rotate(0)}100%{transform:rotate(360deg)}}

const default\_data = {

visible: false,

content: '',

duration: 2,

type: 'default', // default || success || warning || error

};

let timmer = null;

Component({

externalClasses: ,

data: {

...default\_data

},

methods: {

handleShow (options) {

const { type = 'default', duration = 2 } = options;

this.setData({

...options,

type,

duration,

visible: true

});

const d = this.data.duration \* 1000;

if (timmer) clearTimeout(timmer);

if (d !== 0) {

timmer = setTimeout(() => {

this.handleHide();

timmer = null;

}, d);

}

},

handleHide () {

this.setData({

...default\_data

});

}

}

});<view class="i-modal-mask i-class-mask {{ visible ? 'i-modal-mask-show' : '' }}"></view>

<view class="i-class i-modal {{ visible ? 'i-modal-show' : '' }}">

<view class="i-modal-main">

<view class="i-modal-content">

<view class="i-modal-title" wx:if="{{ title }}">{{ title }}</view>

<view class="i-modal-body"><slot></slot></view>

<view class="i-modal-actions" wx:if="{{ actions.length }}">

<block wx:if="{{ actionMode === 'horizontal' }}">

<i-grid i-class="i-modal-grid">

<i-grid-item i-class="{{ actions.length === (index + 1) ? 'i-modal-grid-item-last' : 'i-modal-grid-item' }}" wx:for="{{ actions }}" wx:key="{{ item.name }}">

<template is="button" data="{{ item, index }}"></template>

</i-grid-item>

</i-grid>

</block>

<block wx:else>

<view class="i-modal-action-vertical" wx:for="{{ actions }}" wx:key="{{ item.name }}">

<template is="button" data="{{ item, index }}"></template>

</view>

</block>

</view>

<view class="i-modal-actions" wx:else>

<i-grid i-class="i-modal-grid" wx:if="{{ showOk || showCancel }}">

<i-grid-item i-class="i-modal-grid-item" wx:if="{{ showCancel }}">

<i-button i-class="i-modal-btn-cancel" long type="ghost" bind:click="handleClickCancel">{{ cancelText }}</i-button>

</i-grid-item>

<i-grid-item i-class="i-modal-grid-item-last" wx:if="{{ showOk }}">

<i-button i-class="i-modal-btn-ok" long type="ghost" bind:click="handleClickOk">{{ okText }}</i-button>

</i-grid-item>

</i-grid>

</view>

</view>

</view>

</view>

<template name="button">

<i-button long type="ghost" data-index="{{ index }}" bind:click="handleClickItem">

<view class="i-modal-btn-loading" wx:if="{{ item.loading }}"></view>

<i-icon wx:if="{{ item.icon }}" type="{{ item.icon }}" i-class="i-modal-btn-icon"></i-icon>

<view class="i-modal-btn-text" style="{{ item.color ? 'color: ' + item.color : '' }}">{{ item.name }}</view>

</i-button>

</template>.i-noticebar{display:flex;height:72rpx;line-height:72rpx;font-size:14px;color:#f76a24;background-color:#fefcec;overflow:hidden}.i-noticebar-icon{display:flex;margin-left:30rpx;align-items:center}.i-noticebar-icon+view{margin-left:10rpx}.i-noticebar-operation{display:flex;margin-right:16rpx;align-items:center}.i-noticebar-content-wrap{position:relative;flex:1;margin:0 30rpx;overflow:hidden;text-overflow:ellipsis;white-space:nowrap}.i-noticebar-content-wrap .i-noticebar-content{position:absolute;transition-duration:20s}

Component({

externalClasses: ,

properties: {

visible: {

type: Boolean,

value: false

},

title: {

type: String,

value: ''

},

showOk: {

type: Boolean,

value: true

},

showCancel: {

type: Boolean,

value: true

},

okText: {

type: String,

value: '确定'

},

cancelText: {

type: String,

value: '取消'

},

// 按钮组，有此值时，不显示 ok 和 cancel 按钮

actions: {

type: Array,

value:

},

// horizontal || vertical

actionMode: {

type: String,

value: 'horizontal'

}

},

methods: {

handleClickItem ({ currentTarget = {} }) {

const dataset = currentTarget.dataset || {};

const { index } = dataset;

this.triggerEvent('click', { index });

},

handleClickOk () {

this.triggerEvent('ok');

},

handleClickCancel () {

this.triggerEvent('cancel');

}

}

});

<view wx:if="{{ show }}" class="i-class i-noticebar" style="color: {{ color }}; background-color: {{ backgroundcolor }}">

<i-icon wx:if="{{ icon }}" type="{{ icon }}" size="24" color="{{color}}" class="i-noticebar-icon"></i-icon>

<view class="i-noticebar-content-wrap">

<view class="i-noticebar-content {{loop?'i-noticebar-content-loop':''}}" animation="{{ animationData }}">

<slot></slot>

</view>

</view>

<i-icon wx:if="{{closable}}" class="i-noticebar-operation" type="close" size="20" color="{{color}}" bindtap="handleClose"></i-icon>

</view>

.i-page{display:block;width:100%;height:44px;overflow:hidden;box-sizing:border-box;position:relative}.i-page-prev{position:absolute;left:10px;top:0}.i-page-next{position:absolute;right:10px;top:0}.i-page-number{width:100%;height:44px;line-height:44px;margin:0 auto;text-align:center}.i-page-number-current{display:inline;color:#2d8cf0}.i-page-pointer{width:100%;height:44px;line-height:44px;margin:0 auto;text-align:center}.i-page-pointer-dot{display:inline-block;width:8px;height:8px;margin:0 2px;border-radius:50%;background:#bbbec4}.i-page-pointer-dot.current{background:#80848f}.i-page-button{display:inline-block;margin:0}

Component({

externalClasses: ,

properties: {

// circle || square

shape: {

type: String,

value: 'circle'

},

// small || large || default

size: {

type: String,

value: 'default'

},

src: {

type: String,

value: ''

}

}

});

<view class="i-class i-avatar i-avatar-{{ shape }} i-avatar-{{ size }} {{ src ? 'i-avatar-image' : '' }}">

<image src="{{ src }}" wx:if="{{ src !== '' }}"></image>

<view class="i-avatar-string" wx:else><slot></slot></view>

</view>.i-avatar{display:inline-block;text-align:center;background:#ccc;color:#fff;white-space:nowrap;position:relative;overflow:hidden;vertical-align:middle;width:32px;height:32px;line-height:32px;border-radius:16px;font-size:18px}.i-avatar .ivu-avatar-string{line-height:32px}.i-avatar-large{width:40px;height:40px;line-height:40px;border-radius:20px;font-size:24px}.i-avatar-large .ivu-avatar-string{line-height:40px}.i-avatar-small{width:24px;height:24px;line-height:24px;border-radius:12px;font-size:14px}.i-avatar-small .ivu-avatar-string{line-height:24px}.i-avatar-image{background:0 0}.i-avatar-square{border-radius:4px}.i-avatar>image{width:100%;height:100%}

const VALID\_MODE = ;

const FONT\_COLOR = '#f60';

const BG\_COLOR = '#fff7cc';

Component({

externalClasses: ,

properties: {

closable: {

type: Boolean,

value: false

},

icon: {

type: String,

value: ''

},

loop: {

type: Boolean,

value: false

},

// 背景颜色

backgroundcolor: {

type: String,

value: '#fefcec'

},

// 字体及图标颜色

color: {

type: String,

value: '#f76a24'

},

// 滚动速度

speed: {

type: Number,

value: 1000

}

},

data: {

show: true,

wrapWidth: 0,

width: 0,

duration: 0,

animation: null,

timer: null,

},

detached() {

this.destroyTimer();

},

ready() {

if (this.data.loop) {

this.initAnimation();

}

},

methods: {

initAnimation() {

wx.createSelectorQuery().in(this).select('.i-noticebar-content-wrap').boundingClientRect((wrapRect) => {

wx.createSelectorQuery().in(this).select('.i-noticebar-content').boundingClientRect((rect) => {

const duration = rect.width / 40 \* this.data.speed;

const animation = wx.createAnimation({

duration: duration,

timingFunction: "linear",

});

this.setData({

wrapWidth: wrapRect.width,

width: rect.width,

duration: duration,

animation: animation

}, () => {

this.startAnimation();

});

}).exec();

}).exec();

},

startAnimation() {

//reset

if (this.data.animation.option.transition.duration !== 0) {

this.data.animation.option.transition.duration = 0;

const resetAnimation = this.data.animation.translateX(this.data.wrapWidth).step();

this.setData({

animationData: resetAnimation.export()

});

}

this.data.animation.option.transition.duration = this.data.duration;

const animationData = this.data.animation.translateX(-this.data.width).step();

setTimeout(() => {

this.setData({

animationData: animationData.export()

});

}, 100);

const timer = setTimeout(() => {

this.startAnimation();

}, this.data.duration);

this.setData({

timer,

});

},

destroyTimer() {

if (this.data.timer) {

clearTimeout(this.data.timer);

}

},

handleClose() {

this.destroyTimer();

this.setData({

show: false,

timer: null

});

}

}

});

<view class="i-class i-page">

<view class="i-page-prev" wx:if="{{ mode === 'button' }}">

<i-button i-class="i-page-button" type="ghost" bindclick="handlePrev" disabled="{{ current === 1 }}"><slot name="prev"></slot></i-button>

</view>

<view class="i-page-number" wx:if="{{ mode !== 'pointer' && !simple }}">

<view class="i-page-number-current">{{ current }}</view>/{{total}}

</view>

<view class="i-page-pointer" wx:if="{{ mode === 'pointer' }}">

<view class="i-page-pointer-dot {{ (index + 1) === current ? 'current' : '' }}" wx:for="{{ total }}" wx:key="index"></view>

</view>

<view class="i-page-next" wx:if="{{ mode === 'button' }}">

<i-button i-class="i-page-button" type="ghost" bindclick="handleNext" disabled="{{ current === total }}"><slot name="next"></slot></i-button>

</view>

</view>

.i-panel{position:relative;overflow:hidden}.i-panel-title{font-size:14px;line-height:1;color:#1c2438;padding:20px 16px 10px}.i-panel-title-hide-top{padding-top:0}.i-panel-content{position:relative;background:#fff;overflow:hidden}.i-panel-content::after{content:'';position:absolute;top:0;left:0;width:200%;height:200%;transform:scale(.5);transform-origin:0 0;pointer-events:none;box-sizing:border-box;border:0 solid #e9eaec;border-top-width:1px;border-bottom-width:1px}.i-panel-without-border::after{border:0 none}

Component({

externalClasses: ,

options: {

multipleSlots: true

},

properties: {

// button || number || pointer

mode: {

type: String,

value: 'button'

},

current: {

type: Number,

value: 1

},

total: {

type: Number,

value: 0

},

// 是否隐藏数值

simple: {

type: Boolean,

value: false

}

},

methods: {

handleChange (type) {

this.triggerEvent('change', {

type: type

});

},

handlePrev () {

this.handleChange('prev');

},

handleNext () {

this.handleChange('next');

}

}

});

<view class="i-class i-panel">

<view wx:if="{{ title }}" class="i-panel-title {{ hideTop ? 'i-panel-title-hide-top' : '' }}">{{ title }}</view>

<view class="i-panel-content {{ hideBorder ? 'i-panel-without-border' : '' }}"><slot></slot></view>

</view>

.i-progress{display:inline-block;width:100%;font-size:12px;position:relative}.i-progress-outer{display:inline-block;width:100%;margin-right:0;padding-right:0;box-sizing:border-box}.i-progress-show-info .i-progress-outer{padding-right:55px;margin-right:-55px}.i-progress-inner{display:inline-block;width:100%;background-color:#f3f3f3;border-radius:100px;vertical-align:middle}.i-progress-bg{border-radius:100px;background-color:#2db7f5;transition:all .2s linear;position:relative}.i-progress-text{display:inline-block;margin-left:5px;text-align:left;font-size:1em;vertical-align:middle}.i-progress-active .i-progress-bg:before{content:'';opacity:0;position:absolute;top:0;left:0;right:0;bottom:0;background:#fff;border-radius:10px;animation:i-progress-active 2s ease-in-out infinite}.i-progress-wrong .i-progress-bg{background-color:#ed3f14}.i-progress-wrong .i-progress-text{color:#ed3f14}.i-progress-success .i-progress-bg{background-color:#19be6b}.i-progress-success .i-progress-text{color:#19be6b}@keyframes i-progress-active{0%{opacity:.3;width:0}100%{opacity:0;width:100%}}

Component({

externalClasses: ,

properties: {

title: {

type: String,

value: ''

},

// 标题顶部距离

hideTop: {

type: Boolean,

value: false

},

hideBorder: {

type: Boolean,

value: false

}

}

});

<view class="i-class i-progress i-progress-{{ status }} {{ !hideInfo ? 'i-progress-show-info' : '' }}">

<view class="i-progress-outer">

<view class="i-progress-inner">

<view class="i-progress-bg" style="width: {{percent}}%;height: {{strokeWidth}}px"></view>

</view>

</view>

<view class="i-progress-text" wx:if="{{ !hideInfo }}">

<view class="i-progress-text-inner">{{ percent }}%</view>

</view>

</view>.i-radio-cell::after{display:block}.i-radio-radio-left{float:left}.i-radio-radio-right{float:right}.i-radio-radio{vertical-align:middle}.i-radio-title{display:inline-block;vertical-align:middle}

Component({

externalClasses: ,

properties: {

percent: {

type: Number,

value: 0

},

// normal || active || wrong || success

status: {

type: String,

value: 'normal'

},

strokeWidth: {

type: Number,

value: 10

},

hideInfo: {

type: Boolean,

value: false

}

}

});

<view class="i-class i-radio" catchtap="radioChange">

<i-cell i-class="i-radio-cell">

<label>

<radio value="{{value}}" checked="{{checked}}" color="{{checked?color:''}}" disabled="{{disabled}}" class="i-radio-radio {{positionCls}}" />

<view class="i-radio-title">{{value}}</view>

</label>

</i-cell>

</view>

const prefixCls = 'i-radio';

Component({

externalClasses: ,

relations: {

'../radio-group/index': {

type: 'parent'

}

},

properties: {

value: {

type: String,

value: ''

},

checked: {

type: Boolean,

value: false

},

disabled: {

type: Boolean,

value: false

},

color: {

type: String,

value: '#2d8cf0'

},

position: {

type: String,

value: 'left', //left right

observer: 'setPosition'

}

},

data: {

checked: true,

positionCls: `${prefixCls}-radio-left`,

},

attached() {

this.setPosition();

},

methods: {

changeCurrent(current) {

this.setData({ checked: current });

},

radioChange() {

if (this.data.disabled) return;

const item = { current: !this.data.checked, value: this.data.value };

const parent = this.getRelationNodes('../radio-group/index');

parent ? parent.emitEvent(item) : this.triggerEvent('change', item);

},

setPosition() {

this.setData({

positionCls: this.data.position.indexOf('left') !== -1 ? `${prefixCls}-radio-left` : `${prefixCls}-radio-right`,

});

}

}

});

<i-cell-group class="i-class">

<slot></slot>

</i-cell-group>

.i-rate{margin:0;padding:0;font-size:20px;display:inline-block;vertical-align:middle;font-weight:400;font-style:normal}.i-rate-hide-input{display:none}.i-rate-star{display:inline-block;color:#e9e9e9}.i-rate-current{color:#f5a623}.i-rate-text{display:inline-block;vertical-align:middle;margin-left:6px;font-size:14px}

Component({

externalClasses: ,

relations: {

'../radio/index': {

type: 'child',

linked() {

this.changeCurrent();

},

linkChanged() {

this.changeCurrent();

},

unlinked() {

this.changeCurrent();

}

}

},

properties: {

current: {

type: String,

value: '',

observer: 'changeCurrent'

},

},

methods: {

changeCurrent(val = this.data.current) {

let items = this.getRelationNodes('../radio/index');

const len = items.length;

if (len > 0) {

items.forEach(item => {

item.changeCurrent(val === item.data.value);

});

}

},

emitEvent(current) {

this.triggerEvent('change', current);

}

}

});

<view class="i-class i-rate"

bindtouchmove="handleTouchMove">

<input type="text" :name="name" wx:value="{{value}}" class="i-rate-hide-input" />

<view

wx:for="{{count}}"

wx:key="{{item}}"

class="i-rate-star {{ parse.getCurrent( value,index ) }}"

data-index="{{index}}"

bindtap="handleClick">

<i-icon type="collection\_fill" size="{{size}}"></i-icon>

</view>

<view class="i-rate-text" wx:if="{{ value !== 0 }}"><slot></slot></view>

</view>

<wxs module="parse">

var prefixCls = 'i-rate';

module.exports = {

getCurrent : function( value,index ){

if( index < value ){

return prefixCls + '-current'

}

}

}

</wxs>

.i-row:after{content:"";display:table;clear:both}

Component({

externalClasses: ,

properties : {

count : {

type : Number,

value : 5

},

value : {

type : Number,

value : 0

},

disabled : {

type : Boolean,

value : false

},

size : {

type : Number,

value : 20

},

name : {

type : String,

value : ''

}

},

data : {

touchesStart : {

pageX : 0

}

},

methods : {

handleClick(e){

const data = this.data;

if( data.disabled ){

return;

}

const index = e.currentTarget.dataset.index;

this.triggerEvent('change',{

index : index + 1

})

},

handleTouchMove(e){

const data = this.data;

if( data.disabled ){

return;

}

if( !e.changedTouches ){

return;

}

const movePageX = e.changedTouches.pageX;

const space = movePageX - data.touchesStart.pageX;

if( space <= 0 ){

return;

}

let setIndex = Math.ceil( space/data.size );

setIndex = setIndex > data.count ? data.count : setIndex ;

this.triggerEvent('change',{

index : setIndex

})

}

},

ready(){

const className = '.i-rate';

var query = wx.createSelectorQuery().in(this)

query.select( className ).boundingClientRect((res)=>{

this.data.touchesStart.pageX = res.left || 0;

}).exec()

}

});

<view class="i-class i-row"><slot></slot></view>

Component({

externalClasses: ,

relations: {

'../col/index': {

type: 'child'

}

}

});

<view class="i-class i-slide" capture-catch:touchstart="handleTap2" capture-catch:touchmove="handleTap3">

1111

</view>.i-spin{color:#2d8cf0;vertical-align:middle;text-align:center}.i-spin-dot{position:relative;display:block;border-radius:50%;background-color:#2d8cf0;width:20px;height:20px;animation:ani-spin-bounce 1s 0s ease-in-out infinite}.i-spin-large .i-spin-dot{width:32px;height:32px}.i-spin-small .i-spin-dot{width:12px;height:12px}.i-spin-fix{position:absolute;top:0;left:0;z-index:8;width:100%;height:100%;background-color:rgba(255,255,255,.9)}.i-spin-fullscreen{z-index:2010}.i-spin-fullscreen-wrapper{position:fixed;top:0;right:0;bottom:0;left:0}.i-spin-fix .i-spin-main{position:absolute;top:50%;left:50%;-ms-transform:translate(-50%,-50%);transform:translate(-50%,-50%)}.i-spin-fix .i-spin-dot{display:inline-block}.i-spin-show-text .i-spin-dot,.i-spin-text{display:none}.i-spin-show-text .i-spin-text{display:block;font-size:14px}@keyframes ani-spin-bounce{0%{transform:scale(0)}100%{transform:scale(1);opacity:0}}

Component({

externalClasses: ,

options: {

// 在组件定义时的选项中启用多slot支持

multipleSlots: true

},

methods : {

handleTap2(){

console.log(event,1111111)

},

handleTap3(){

}

}

});

<view class="i-class i-spin i-spin-{{ size }} {{ fix ? 'i-spin-fix' : '' }} {{ custom ? 'i-spin-show-text' : '' }} {{ fullscreen ? 'i-spin-fullscreen' : '' }}">

<div class="i-spin-main">

<view class="i-spin-dot"></view>

<div class="i-spin-text"><slot></slot></div>

</div>

</view>

.i-step-ico{width:24px;height:100%;border-radius:100%;background:#fff;position:relative;z-index:2;margin:0 auto;border:#dddee1 solid 1px}.i-step-ico-in{vertical-align:baseline}.i-step-line{position:absolute;left:50%;top:12px;width:100%;height:1px;background:#dddee1}.i-step-horizontal .i-step-ico::after{position:absolute;top:11px;left:23px;z-index:1;content:'';height:1px;background:#fff;width:10px}.i-step-horizontal .i-step-item-main{text-align:center}.i-step-horizontal .i-step-ico::before{position:absolute;top:11px;left:-11px;z-index:1;content:'';height:1px;background:#fff;width:10px}.i-step-ico{box-sizing:border-box;font-size:12px}.i-step-process .i-step-ico{border:#2d8cf0 solid 1px;color:#fff;background:#2d8cf0}.i-step-wait .i-step-ico{border:#e9eaec solid 1px;color:#e9eaec}.i-step-wait .i-step-line{background:#2d8cf0}.i-step-finish .i-step-ico{border:#2d8cf0 solid 1px;color:#2d8cf0}.i-step-finish .i-step-line{background:#2d8cf0}.i-step-error .i-step-ico{border:#ed3f14 solid 1px;color:#ed3f14}.i-step-error .i-step-line{background:#ed3f14}.i-step-item{font-size:12px;position:relative;display:inline-block;box-sizing:border-box;padding-left:10px;vertical-align:top}.i-step-item-ico{width:100%;height:24px;line-height:24px;text-align:center}.i-step-item-main{margin-top:10px;clear:both}.i-step-item-title{font-size:14px;font-weight:700;color:#1c2438}.i-step-item-content{font-size:12px;font-weight:700;margin-top:2px;color:#80848f}.i-step-vertical{padding-bottom:30px}.i-step-vertical .i-step-item-ico{width:24px;float:left}.i-step-vertical .i-step-item-main{margin-left:40px;margin-top:0;clear:inherit}.i-step-vertical .i-step-line{position:absolute;height:100%;top:0;left:10px;margin:0 0 0 12px;width:1px}

Component({

externalClasses: ,

properties: {

// small || default || large

size: {

type: String,

value: 'default'

},

fix: {

type: Boolean,

value: false

},

fullscreen: {

type: Boolean,

value: false

},

custom: {

type: Boolean,

value: false

}

}

});

<view class="i-class i-step-item {{parse.getClass(status,current,index)}} {{ direction === 'vertical' ? 'i-step-vertical' : 'i-step-horizontal' }}" style="{{parse.getItemStyle(len,direction)}}">

<view class="i-step-item-ico">

<view class="i-step-ico" wx:if="{{parse.noIco(status,current,index,icon) }}">{{ index+1 }}</view>

<view class="i-step-ico" wx:else>

<i-icon i-class="i-step-ico-in" type="{{parse.getIcoClass(status,icon)}}"></i-icon>

</view>

<view class="i-step-line" wx:if="{{ index !== len - 1 }}"></view>

</view>

<view class="i-step-item-main">

<view class="i-step-item-title" wx:if="{{title !== ''}}">

{{title}}

</view>

<view class="i-step-item-title" wx:else>

<slot name="title"></slot>

</view>

<view class="i-step-item-content" wx:if="{{content !== ''}}">

{{content}}

</view>

<view class="i-step-item-content" wx:else>

<slot name="content"></slot>

</view>

</view>

</view>

<wxs module="parse">

var allStatus = ;

module.exports = {

noIco : function( status,current,index,icon ){

var aindex = allStatus.indexOf(status);

var noIcon = true;

if( index < current || icon !== '' ){

noIcon = false;

}

return noIcon;

},

getIcoClass : function( status,ico ){

var class = '';

if( status === 'error' ){

class = 'close';

}else{

class = 'right';

}

if( ico !== '' ){

class = ico;

}

return class;

},

getItemStyle : function(len,direction){

if( direction === 'horizontal' ){

return 'width :'+100/len + '%';

}else{

return 'width : 100%;';

}

},

getClass : function( status,current,index ) {

//wait、process、finish、error

var startClass = 'i-step-'

var classes = '';

var cindex = allStatus.indexOf( status );

if( cindex !== -1 ){

classes = startClass + allStatus;

}

if( index < current ){

classes = startClass + 'finish';

}else if( index === current ){

classes = startClass + 'process';

}

return classes;

}

}

</wxs>.i-steps{width:100%}

Component({

externalClasses: ,

properties: {

count: {

type: Number,

value: 0,

observer: 'finalCount'

},

overflowCount: {

type: Number,

value: 99

},

dot: {

type: Boolean,

value: false

},

},

data: {

finalCount: 0

},

methods: {

finalCount() {

this.setData({

finalCount: parseInt(this.data.count) >= parseInt(this.data.overflowCount) ? `${this.data.overflowCount}+` : this.data.count

});

},

}

});

<view class="i-class i-badge">

<slot></slot>

<view class="i-badge-dot" wx:if="{{ dot }}"></view>

<view class="i-badge-count i-class-alone" wx:elif="{{ count !== 0 }}">{{ finalCount }}</view>

</view>

.i-badge{position:relative;display:inline-block;line-height:1;vertical-align:middle}.i-badge-count{position:absolute;transform:translateX(50%);top:-6px;right:0;height:18px;border-radius:9px;min-width:18px;background:#ed3f14;border:1px solid transparent;color:#fff;line-height:18px;text-align:center;padding:0 5px;font-size:12px;white-space:nowrap;transform-origin:-10% center;z-index:10;box-shadow:0 0 0 1px #fff;box-sizing:border-box;text-rendering:optimizeLegibility}.i-badge-count-alone{top:auto;display:block;position:relative;transform:translateX(0)}.i-badge-dot{position:absolute;transform:translateX(-50%);transform-origin:0 center;top:-4px;right:-8px;height:8px;width:8px;border-radius:100%;background:#ed3f14;z-index:10;box-shadow:0 0 0 1px #fff}

Component({

externalClasses: ,

properties : {

status : {

type : String,

//wait、process、finish、error

value : ''

},

title : {

type : String,

value : ''

},

content : {

type : String,

value : ''

},

icon : {

type : String,

value : ''

}

},

options: {

// 在组件定义时的选项中启用多slot支持

multipleSlots: true

},

relations : {

'../steps/index' : {

type : 'parent'

}

},

data : {

//step length

len : 1,

//current in step index

index : 0,

//parent component select current index

current : 0,

//css direction

direction : 'horizontal'

},

methods : {

updateDataChange( options ){

this.setData({

len : options.len,

index : options.index,

current : options.current,

direction : options.direction

})

}

}

})<view class="i-class i-steps">

<slot></slot>

</view>

Component({

externalClasses: ,

properties : {

current : {

type : Number,

value : -1,

observer : '\_updateDataChange'

},

status : {

type : String,

//wait、process、finish、error

value : ''

},

direction : {

type : String,

//value has horizontal or vertical

value : 'horizontal'

}

},

relations : {

'../step/index' : {

type : 'child',

linked(){

this.\_updateDataChange();

},

linkChanged () {

this.\_updateDataChange();

},

unlinked () {

this.\_updateDataChange();

}

}

},

methods: {

\_updateDataChange() {

let steps = this.getRelationNodes('../step/index');

const len = steps.length;

if (len > 0) {

steps.forEach((step, index) => {

step.updateDataChange({

len : len,

index : index,

current : this.data.current,

direction : this.data.direction

});

});

}

}

}

})<view class="i-sticky i-class">

<slot></slot>

</view>

.i-sticky-item-header{background:#eee;font-size:14px;width:100%;height:32px;line-height:32px}.i-sticky-item-content{font-size:14px}.i-sticky-title{width:100%;padding:0 15px;box-sizing:border-box;background:#eee}.i-sticky-fixed .i-sticky-title{position:fixed;top:0}

Component({

externalClasses: ,

properties : {

scrollTop : {

type : Number,

observer(val){

this.\_updateScrollTopChange();

}

}

},

relations : {

'../sticky-item/index' : {

type : 'child',

linked(){

this.\_updateDataChange();

},

linkChanged () {

this.\_updateDataChange();

},

unlinked () {

this.\_updateDataChange();

}

}

},

data : {

timer : null,

itemLength : 0,

},

methods : {

\_updateScrollTopChange(){

const stickies = this.getRelationNodes('../sticky-item/index');

if( stickies.length > 0 ){

stickies.forEach((item) => {

if( item ){

item.updateScrollTopChange( this.data.scrollTop );

}

})

}

},

\_updateDataChange( ){

const stickies = this.getRelationNodes('../sticky-item/index');

if( stickies.length > 0 ){

if( this.data.timer ){

clearTimeout( this.data.timer )

this.setData({

timer : null

})

}

this.data.timer = setTimeout(()=>{

stickies.forEach((item,index) => {

if( item ){

item.updateDataChange(index);

}

})

},0)

this.setData({

timer : this.data.timer

})

}

}

}

})<view class="i-sticky-item">

<view class="i-sticky-item-header i-class {{ isFixed === true ? 'i-sticky-fixed' : '' }}">

<view class="i-sticky-title">

<slot name="title"></slot>

</view>

</view>

<view class="i-index-item-content">

<slot name="content"></slot>

</view>

</view>

.i-swipeout-wrap{border-bottom:#dddee1 solid 1px;background:#fff;position:relative;overflow:hidden}.i-swipeout-item{width:100%;padding:15px 20px;box-sizing:border-box;transition:transform .2s ease;font-size:14px}.i-swipeout-content{white-space:nowrap;overflow:hidden}.i-swipeout-button-right-group{position:absolute;right:-100%;top:0;height:100%;z-index:1;width:100%}.i-swipeout-button-right-item{height:100%;float:left;white-space:nowrap;box-sizing:border-box;display:flex;align-items:center;justify-content:center}

Component({

externalClasses: ,

options: {

multipleSlots: true

},

relations : {

'../sticky/index' : {

type : 'parent'

}

},

data : {

top : 0,

height : 0,

isFixed : false,

index : -1,

},

methods: {

updateScrollTopChange(scrollTop){

const data = this.data;

const top = data.top;

const height = data.height;

this.setData({

isFixed : ( scrollTop >= top && scrollTop < top + height ) ? true : false

})

},

updateDataChange(index) {

const className = '.i-sticky-item';

const query = wx.createSelectorQuery().in(this);

query.select( className ).boundingClientRect((res)=>{

if( res ){

this.setData({

top : res.top,

height : res.height,

index : index

})

}

}).exec()

}

}

})<view class="i-swipeout-wrap i-class">

<view class="i-swipeout-item" bindtouchstart="handlerTouchstart" bindtouchmove="handlerTouchmove" bindtouchend="handlerTouchend" style="{{parse.setPosition(position)}}">

<view class="i-swipeout-content">

<slot name="content"></slot>

</view>

<view class="i-swipeout-button-right-group"

wx:if="{{actions.length > 0}}"

catchtouchend="loop" >

<view class="i-swipeout-button-right-item"

wx:for="{{actions}}"

style="{{parse.setStyle(item)}};width:{{item.width}}px;"

wx:key="{{index}}"

data-index="{{index}}"

bind:tap="handlerButton">

<i-icon

wx:if="{{item.icon}}"

type="{{item.icon}}"

size="{{item.fontsize}}"

color="{{item.color}}">

</i-icon>

{{item.name}}

</view>

</view>

<view class="i-swipeout-button-right-group" catchtouchend="loop" bind:tap="handlerParentButton" wx:if="{{actions.length === 0}}" style="width:{{operateWidth}}px;right:-{{operateWidth}}px;">

<slot name="button"></slot>

</view>

</view>

</view>

<wxs module="parse">

module.exports = {

setStyle : function( item ){

var defaults = '#f7f7f7';

return 'background:' + ( item.background ? item.background : defaults ) +';' + 'color:'+ item.color;

},

setPosition : function( position ){

return 'transform:translate(' + position.pageX + 'px,0);';

}

}

</wxs>.i-switch{display:inline-block;width:48px;height:24px;line-height:24px;border-radius:24px;vertical-align:middle;border:1px solid #ccc;background-color:#ccc;position:relative;cursor:pointer;-webkit-tap-highlight-color:transparent;transition:all .2s ease-in-out}.i-switch-hide-input{display:none;opacity:0}.i-switch-inner{color:#fff;font-size:12px;position:absolute;left:25px;vertical-align:middle}.i-switch-inner .i-icon{width:12px;height:12px;text-align:center;vertical-align:middle}.i-switch:after{content:'';width:22px;height:22px;border-radius:22px;background-color:#fff;position:absolute;left:1px;top:1px;cursor:pointer;transition:left .2s ease-in-out,width .2s ease-in-out}.i-switch-checked:after{left:8px}.i-switch-large{width:60px}.i-switch-large.i-switch-checked:after{left:37px}.i-switch-checked:after{left:25px}.i-switch-checked{border-color:#2d8cf0;background-color:#2d8cf0}.i-switch-checked .i-switch-inner{left:8px}.i-switch-checked:after{left:25px}.i-switch-disabled{background:#f3f3f3;border-color:#f3f3f3}.i-switch-disabled:after{background:#ccc;cursor:not-allowed}.i-switch-disabled .i-switch-inner{color:#ccc}

/\*

\* touch事件判断方式

\* https://github.com/madrobby/zepto/blob/master/src/touch.js#files

\*/

function swipeDirection(x1, x2, y1, y2) {

return Math.abs(x1 - x2) >=

Math.abs(y1 - y2) ? (x1 - x2 > 0 ? 'Left' : 'Right') : (y1 - y2 > 0 ? 'Up' : 'Down')

}

Component({

externalClasses: ,

properties: {

actions: {

value: ,

type: Array,

observer : '\_updateButtonSize'

},

unclosable : {

value : false,

type : Boolean

},

toggle : {

value : false,

type : Boolean,

observer : 'closeButtonGroup'

},

operateWidth : {

type : Number,

value : 160

}

},

options: {

// 在组件定义时的选项中启用多slot支持

multipleSlots: true

},

data : {

//touch start position

tStart : {

pageX : 0,

pageY : 0

},

//限制滑动距离

limitMove : 0,

//element move position

position : {

pageX : 0,

pageY : 0

}

},

methods : {

//阻止事件冒泡

loop(){},

\_updateButtonSize(){

const actions = this.data.actions;

if( actions.length > 0 ){

const query = wx.createSelectorQuery().in(this);

let limitMovePosition = 0;

actions.forEach(item => {

limitMovePosition += item.width || 0;

});

this.data.limitMove = limitMovePosition;

/\*

\* 动态获取每个传进值的按钮尺寸不能正确获取，在安卓上少了6px

\* 暂时实现需要在actions里面传递宽度

\* 需要后期调研

\*/

//query.selectAll('.i-swipeout-button-right-item').boundingClientRect((rects)=>{

// if( rects ){

// console.log(rects,1111111)

// rects.forEach(item => {

// limitMovePosition += item.width;

// });

// this.data.limitMove = limitMovePosition;

// console.log(limitMovePosition,111111111)

// }

// }).exec()

}else{

this.data.limitMove = this.data.operateWidth;

}

},

handlerTouchstart(event){

const touches = event.touches ? event.touches : {};

const tStart = this.data.tStart;

if( touches ){

for( let i in tStart ){

if( touches ){

tStart = touches;

}

}

}

},

swipper(touches){

const data = this.data;

const start = data.tStart;

const spacing = {

pageX : touches.pageX - start.pageX,

pageY : touches.pageY - start.pageY

}

if( data.limitMove < Math.abs( spacing.pageX ) ){

spacing.pageX = -data.limitMove;

}

this.setData({

'position' : spacing

})

},

handlerTouchmove(event){

const start = this.data.tStart;

const touches = event.touches ? event.touches : {};

if( touches ){

const direction = swipeDirection( start.pageX,touches.pageX,start.pageY,touches.pageY );

if( direction === 'Left' ){

this.swipper( touches );

}

}

},

handlerTouchend(event){

const start = this.data.tStart;

const touches = event.changedTouches ? event.changedTouches : {};

if( touches ){

const direction = swipeDirection( start.pageX,touches.pageX,start.pageY,touches.pageY );

const spacing = {

pageX : touches.pageX - start.pageX,

pageY : touches.pageY - start.pageY

}

if( Math.abs( spacing.pageX ) >= 40 && direction === "Left" ){

spacing.pageX = spacing.pageX < 0 ? - this.data.limitMove : this.data.limitMove;

}else{

spacing.pageX = 0;

}

this.setData({

'position' : spacing

})

}

},

handlerButton(event){

if( !this.data.unclosable ){

this.closeButtonGroup();

}

const dataset = event.currentTarget.dataset;

this.triggerEvent('change',{

index : dataset.index

})

},

closeButtonGroup(){

this.setData({

'position' : {pageX : 0,pageY : 0}

})

},

//控制自定义组件

handlerParentButton(event){

if( !this.data.unclosable ){

this.closeButtonGroup();

}

}

},

ready(){

this.\_updateButtonSize();

}

});

<view class="i-class i-switch {{parse.setSize(size)}} {{parse.setCurrent(value,disabled)}}" bindtap="toggle">

<input type="text" :name="name" wx:value="{{value}}" class="i-switch-hide-input"></input>

<view class="i-switch-inner" wx:if="{{value === true}}">

<slot name="open"></slot>

</view>

<view class="i-switch-inner" wx:else>

<slot name="close"></slot>

</view>

</view>

<wxs module="parse">

var sizes = ;

var prefixCls = 'i-switch';

module.exports = {

setSize : function( size ){

var index = sizes.indexOf( size );

return prefixCls + ( index > -1 ? ( '-'+size ) : 'default' )

},

setCurrent : function( value,disabled ){

var className = value && !disabled ? prefixCls + '-checked' : '';

if( disabled ){

className += ' ' + prefixCls + '-disabled';

}

return className;

}

}

</wxs>.i-tabs-tab{flex:1;display:flex;width:100%;-webkit-box-pack:center;justify-content:center;-webkit-box-align:center;align-items:center;-webkit-box-orient:vertical;-webkit-box-direction:normal;flex-direction:column;text-align:center;position:relative}.i-tabs-tab-bar{display:block;width:100%;height:2px;background:0 0;position:absolute;bottom:0;left:0;background:#2d8cf0}.i-tabs-tab-title{font-size:14px;text-align:center;box-sizing:border-box;color:#80848f}.i-tabs-tab-title-current{color:#2d8cf0}.i-tabs-tab-scroll{display:inline-block;width:60px}

Component({

externalClasses: ,

properties : {

value : {

type : Boolean,

value : false

},

//large small default

size : {

type : String,

value : 'default'

},

// is or not disable

disabled : {

type : Boolean,

value : false

},

// hidden inut name

name : {

type : String,

value : ''

}

},

options: {

// 在组件定义时的选项中启用多slot支持

multipleSlots: true

},

methods : {

toggle(){

if( this.data.disabled ) return;

const data = this.data;

const value = data.value ? false : true;

this.triggerEvent('change',{

value : value

})

}

}

});

<view class="i-class i-tabs-tab {{ scroll ? 'i-tabs-tab-scroll' : '' }} {{ current ? 'i-tabs-tab-current' : '' }}">

<i-badge dot="{{ dot }}" count="{{ dot ? 0 : count }}">

<view bindtap="handleClickItem">

<view class="i-tabs-tab-title {{ current ? 'i-tabs-tab-title-current' : '' }}" wx:if="{{ current && currentColor }}" style="color: {{ currentColor }}">{{ title }}</view>

<view class="i-tabs-tab-title {{ current ? 'i-tabs-tab-title-current' : '' }}" wx:else>{{ title }}</view>

</view>

</i-badge>

<view class="i-tabs-tab-bar" wx:if="{{ current }}" style="background: {{ currentColor }}"></view>

</view>.i-tab-bar{display:flex;width:100%;height:50px;box-sizing:border-box;position:relative;justify-content:space-around;align-items:center;-webkit-box-align:center;background:#fff}.i-tab-bar::after{content:'';position:absolute;top:0;left:0;width:200%;height:200%;transform:scale(.5);transform-origin:0 0;pointer-events:none;box-sizing:border-box;border:0 solid #e9eaec;border-top-width:1px}.i-tab-bar-fixed{position:fixed;bottom:0;z-index:2}.i-tab-bar-list{position:absolute;top:0;bottom:0;left:0;right:0}.i-tab-bar-layer{display:block;float:left;height:100%}

Component({

externalClasses: ,

relations: {

'../tabs/index': {

type: 'parent'

}

},

properties: {

key: {

type: String,

value: ''

},

title: {

type: String,

value: ''

},

dot: {

type: Boolean,

value: false

},

count: {

type: Number,

value: 0

}

},

data: {

current: false,

currentColor: '',

scroll: false

},

methods: {

changeCurrent (current) {

this.setData({ current });

},

changeCurrentColor (currentColor) {

this.setData({ currentColor });

},

changeScroll (scroll) {

this.setData({ scroll });

},

handleClickItem () {

const parent = this.getRelationNodes('../tabs/index');

parent.emitEvent(this.data.key);

}

}

});

<view class="i-class i-tab-bar {{ fixed ? 'i-tab-bar-fixed' : '' }}">

<slot></slot>

<view class="i-tab-bar-list">

<view class="i-tab-bar-layer" wx:for="{{ list }}" wx:key="{{ item.key }}" data-key="{{ item.key }}" bindtap="handleClickItem" style="width: {{ 100 / list.length }}%"></view>

</view>

</view>.i-tab-bar-item{flex:1;display:flex;width:100%;-webkit-box-pack:center;justify-content:center;-webkit-box-align:center;align-items:center;-webkit-box-orient:vertical;-webkit-box-direction:normal;flex-direction:column;text-align:center}.i-tab-bar-item-icon{display:flex;-webkit-box-pack:center;justify-content:center;box-sizing:border-box;color:#80848f}.i-tab-bar-item-icon-current{color:#2d8cf0}.i-tab-bar-item-img{display:flex;-webkit-box-pack:center;justify-content:center;box-sizing:border-box;width:22px;height:22px}.i-tab-bar-item-title{font-size:10px;margin:3px 0 0;line-height:1;text-align:center;box-sizing:border-box;color:#80848f}.i-tab-bar-item-title-current{color:#2d8cf0}.i-tab-bar-item-img{display:flex;-webkit-box-pack:center;justify-content:center;box-sizing:border-box;color:#80848f}

Component({

externalClasses: ,

relations: {

'../tab-bar-item/index': {

type: 'child',

linked () {

this.changeCurrent();

},

linkChanged () {

this.changeCurrent();

},

unlinked () {

this.changeCurrent();

}

}

},

properties: {

current: {

type: String,

value: '',

observer: 'changeCurrent'

},

color: {

type: String,

value: ''

},

fixed: {

type: Boolean,

value: false

}

},

data: {

list:

},

methods: {

changeCurrent (val = this.data.current) {

let items = this.getRelationNodes('../tab-bar-item/index');

const len = items.length;

if (len > 0) {

const list = ;

items.forEach(item => {

item.changeCurrent(item.data.key === val);

item.changeCurrentColor(this.data.color);

list.push({

key: item.data.key

});

});

this.setData({

list: list

});

}

},

emitEvent (key) {

this.triggerEvent('change', { key });

},

handleClickItem (e) {

const key = e.currentTarget.dataset.key;

this.emitEvent(key);

}

}

});

<view class="i-class i-tab-bar-item">

<i-badge dot="{{ dot }}" count="{{ dot ? 0 : count }}">

<view>

<i-icon wx:if="{{ icon || currentIcon }}" i-class="i-tab-bar-item-icon {{ current ? 'item-index--i-tab-bar-item-icon-current' : '' }}" color="{{ current ? currentColor : '' }}" type="{{ current ? currentIcon : icon }}" size="22"></i-icon>

<image wx:else class="i-tab-bar-item-img" src="{{ current ? currentImg : img }}"></image>

<view class="i-tab-bar-item-title {{ current ? 'i-tab-bar-item-title-current' : '' }}" wx:if="{{ current && currentColor }}" style="color: {{ currentColor }}">{{ title }}</view>

<view class="i-tab-bar-item-title {{ current ? 'i-tab-bar-item-title-current' : '' }}" wx:else>{{ title }}</view>

</view>

</i-badge>

</view>.i-tabs{display:flex;width:100%;height:42px;line-height:42px;box-sizing:border-box;position:relative;justify-content:space-around;align-items:center;-webkit-box-align:center;background:#fff}.i-tabs::after{content:'';position:absolute;top:0;left:0;width:200%;height:200%;transform:scale(.5);transform-origin:0 0;pointer-events:none;box-sizing:border-box;border:0 solid #e9eaec;border-bottom-width:1px}.i-tabs-scroll{display:block;overflow-x:auto;white-space:nowrap}.i-tabs-fixed{position:fixed;top:0;z-index:2}

Component({

externalClasses: ,

relations: {

'../tab-bar/index': {

type: 'parent'

}

},

properties: {

icon: {

type: String,

value: ''

},

currentIcon: {

type: String,

value: ''

},

img: {

type: String,

value: ''

},

currentImg: {

type: String,

value: ''

},

key: {

type: String,

value: ''

},

title: {

type: String,

value: ''

},

dot: {

type: Boolean,

value: false

},

count: {

type: Number,

value: 0

}

},

data: {

current: false,

currentColor: ''

},

methods: {

changeCurrent (current) {

this.setData({ current });

},

changeCurrentColor (currentColor) {

this.setData({ currentColor });

},

handleClickItem () {

const parent = this.getRelationNodes('../tab-bar/index');

parent.emitEvent(this.data.key);

}

}

});

<scroll-view wx:if="{{ scroll }}" scroll-x="true" class="i-class i-tabs i-tabs-scroll {{ fixed ? 'i-tabs-fixed' : '' }}"><slot></slot></scroll-view>

<view wx:else class="i-class i-tabs {{ fixed ? 'i-tabs-fixed' : '' }}"><slot></slot></view>.i-tag{display:inline-block;height:18px;line-height:18px;padding:0 4px;border-radius:2px;background:#fff;font-size:11px;vertical-align:middle;border:1rpx solid #dddee1}.i-tag-none{border-color:#fff}.i-tag-default{border-color:#dddee1;background:#e9eaec}.i-tag-red{background:#ed3f14;color:#fff}.i-tag-red-border{color:#ed3f14;background:#fff;border-color:#ed3f14}.i-tag-red-checked{background:#ed3f14;color:#fff;border-color:#ed3f14}.i-tag-green{background:#19be6b;color:#fff;border-color:#19be6b}.i-tag-green-border{color:#19be6b;background:#fff;border-color:#19be6b}.i-tag-green-checked{background:#19be6b;color:#fff;border-color:#19be6b}.i-tag-blue{background:#2d8cf0;color:#fff;border-color:#2d8cf0}.i-tag-blue-border{color:#2d8cf0;background:#fff;border-color:#2d8cf0}.i-tag-blue-checked{background:#2d8cf0;color:#fff;border-color:#2d8cf0}.i-tag-yellow{background:#f90;color:#fff;border-color:#f90}.i-tag-yellow-border{color:#f90;background:#fff;border-color:#f90}.i-tag-yellow-checked{background:#f90;color:#fff;border-color:#f90}.i-tag-default-checked{background:#e9eaec;color:#495060;border-color:#e9eaec}

Component({

externalClasses: ,

relations: {

'../tab/index': {

type: 'child',

linked () {

this.changeCurrent();

},

linkChanged () {

this.changeCurrent();

},

unlinked () {

this.changeCurrent();

}

}

},

properties: {

current: {

type: String,

value: '',

observer: 'changeCurrent'

},

color: {

type: String,

value: ''

},

scroll: {

type: Boolean,

value: false

},

fixed: {

type: Boolean,

value: false

}

},

methods: {

changeCurrent (val = this.data.current) {

let items = this.getRelationNodes('../tab/index');

const len = items.length;

if (len > 0) {

items.forEach(item => {

item.changeScroll(this.data.scroll);

item.changeCurrent(item.data.key === val);

item.changeCurrentColor(this.data.color);

});

}

},

emitEvent (key) {

this.triggerEvent('change', { key });

}

}

});

<view

class="i-class i-tag {{ parse.getClass(color,type,checked,checkable) }} {{checkable ? '' : 'i-tag-disable'}}"

bindtap="tapTag">

<slot></slot>

</view>

<wxs module="parse">

module.exports = {

getClass : function(color,type,checked,checkable) {

var initColorList = ;

var theme = '';

var className = 'i-tag-';

if( initColorList.indexOf( color ) > -1 ){

theme = className + color;

}

if( type === 'border' ){

theme = className+color+'-border';

}

if( checkable && checked ){

theme = className+color+'-checked';

}else if( checkable && !checked ){

theme = ( type === 'border' ? className + color +'-border' : className+'none' );

}

return theme;

}

}

</wxs>.i-toast{position:fixed;top:35%;left:50%;transform:translate3d(-50%,-50%,0);background:rgba(0,0,0,.7);color:#fff;font-size:14px;line-height:1.5em;margin:0 auto;box-sizing:border-box;padding:10px 18px;text-align:center;border-radius:4px;z-index:1010}.i-toast-mask{position:fixed;top:0;bottom:0;left:0;right:0;z-index:1010}.i-toast-icon{font-size:38px!important;margin-bottom:6px}.i-toast-image{max-width:100px;max-height:100px}.i-toast-loading{display:inline-block;vertical-align:middle;width:28px;height:28px;background:0 0;border-radius:50%;border:2px solid #fff;border-color:#fff #fff #fff #2d8cf0;animation:btn-spin .8s linear;animation-iteration-count:infinite}@keyframes btn-spin{0%{transform:rotate(0)}100%{transform:rotate(360deg)}}

function getCtx (selector) {

const pages = getCurrentPages();

const ctx = pages;

const componentCtx = ctx.selectComponent(selector);

if (!componentCtx) {

console.error('无法找到对应的组件，请按文档说明使用组件');

return null;

}

return componentCtx;

}

function Toast(options) {

const { selector = '#toast' } = options;

const ctx = getCtx(selector);

ctx.handleShow(options);

}

Toast.hide = function (selector = '#toast') {

const ctx = getCtx(selector);

ctx.handleHide();

};

function Message(options) {

const { selector = '#message' } = options;

const ctx = getCtx(selector);

ctx.handleShow(options);

}

module.exports = {

$Toast: Toast,

$Message: Message

};<button

class="i-class i-btn {{ long ? 'i-btn-long' : '' }} {{ 'i-btn-' + size }} {{ 'i-btn-' + type }} {{ 'i-btn-' + shape }} {{ loading ? 'i-btn-loading' : '' }} {{ disabled ? 'i-btn-disabled' : ''}} {{ inline ? 'i-btn-inline' : '' }}"

hover-class="i-btn-hover"

bindtap="handleTap"

open-type="{{ openType }}"

app-parameter="{{ appParameter }}"

hover-stop-propagation="{{ hoverStopPropagation }}"

hover-start-time="{{ hoverStartTime }}"

hover-stay-time="{{ hoverStayTime }}"

session-from="{{ sessionFrom }}"

send-message-title="{{ sendMessageTitle }}"

send-message-path="{{ sendMessagePath }}"

send-message-img="{{ sendMessageImg }}"

show-message-card="{{ showMessageCard }}"

bindcontact="bindcontact"

bindgetuserinfo="bindgetuserinfo"

bindgetphonenumber="bindgetphonenumber"

binderror="binderror"

plain="true"

><view class="i-btn-loading-inner" wx:if="{{loading}}"></view><slot></slot></button>.i-btn{text-align:center;vertical-align:middle;touch-action:manipulation;cursor:pointer;background-image:none;white-space:nowrap;user-select:none;font-size:14px;border-radius:2px;border:0!important;position:relative;text-decoration:none;height:44px;line-height:44px;box-shadow:inset 0 0 0 1px rgba(0,0,0,.1);color:#fff!important;background:#f7f7f7!important;color:#495060!important;margin:10px}.i-btn-hover{opacity:.9}.i-btn-long{border-radius:0;margin:0;box-shadow:none}.i-btn-large{height:48px;line-height:48px}.i-btn-small{height:40px;line-height:40px}.i-btn-primary{color:#fff!important;background:#2d8cf0!important}.i-btn-ghost{color:#fff!important;background:#fff!important;color:#495060!important}.i-btn-success{color:#fff!important;background:#19be6b!important}.i-btn-warning{color:#fff!important;background:#f90!important}.i-btn-error{color:#fff!important;background:#ed3f14!important}.i-btn-info{color:#fff!important;background:#2db7f5!important}.i-btn-circle{border-radius:44px}.i-btn-large.i-btn-circle{border-radius:48px}.i-btn-small.i-btn-circle{border-radius:40px}.i-btn-loading{opacity:.6}.i-btn-loading-inner{display:inline-block;margin-right:12px;vertical-align:middle;width:14px;height:14px;background:0 0;border-radius:50%;border:2px solid #fff;border-color:#fff #fff #fff transparent;animation:btn-spin .6s linear;animation-iteration-count:infinite}.i-btn-disabled{color:#bbbec4!important;background:#f7f7f7!important}.i-btn-inline{display:inline-block}@keyframes btn-spin{0%{transform:rotate(0)}100%{transform:rotate(360deg)}}

Component({

externalClasses: ,

properties : {

//slot name

name : {

type : String,

value : ''

},

//can click or not click

checkable : {

type : Boolean,

value : false

},

//is current choose

checked : {

type : Boolean,

value : true

},

//background and color setting

color : {

type : String,

value : 'default'

},

//control fill or not

type : {

type : String,

value : 'dot'

}

},

methods : {

tapTag(){

const data = this.data;

if( data.checkable ){

const checked = data.checked ? false : true;

this.triggerEvent('change',{

name : data.name || '',

checked : checked

});

}

}

}

})<view class="i-toast-mask" wx:if="{{ visible && mask }}" bindtap="handleHide"></view>

<view class="i-class i-toast" wx:if="{{ visible }}">

<block wx:if="{{ type !== 'default' }}">

<view class="i-toast-type">

<i-icon i-class="i-toast-icon" type="success" wx:if="{{ type === 'success' }}"></i-icon>

<i-icon i-class="i-toast-icon" type="prompt" wx:elif="{{ type === 'warning' }}"></i-icon>

<i-icon i-class="i-toast-icon" type="delete" wx:elif="{{ type === 'error' }}"></i-icon>

<view class="i-toast-icon i-toast-loading" wx:elif="{{ type === 'loading' }}"></view>

</view>

</block>

<block wx:else>

<i-icon i-class="i-toast-icon" type="{{ icon }}" wx:if="{{ icon }}"></i-icon>

<image class="i-toast-image" src="{{ image }}" wx:if="{{ image }}" mode="aspectFit"></image>

</block>

<view class="i-toast-content" wx:if="{{ content }}">{{ content }}</view>

</view>/\*\*index.wxss\*\*/

page {

background: #f6f6f6;

display: flex;

flex-direction: column;

justify-content: flex-start;

}

.content {

height: 100%;

}

.container {

height: 100%;

top: 0;

bottom: 0;

left: 0;

right: 0;

padding-top: 0rpx;

padding-bottom: 0rpx;

}

.video\_src{

/\* height: 100%; \*/

width: 100%;

}

.section{

padding: 30rpx 0rpx 0rpx 0rpx;

}

const default\_data = {

visible: false,

content: '',

icon: '',

image: '',

duration: 2,

mask: true,

type: 'default', // default || success || warning || error || loading

};

let timmer = null;

Component({

externalClasses: ,

data: {

...default\_data

},

methods: {

handleShow (options) {

const { type = 'default', duration = 2 } = options;

this.setData({

...options,

type,

duration,

visible: true

});

const d = this.data.duration \* 1000;

if (timmer) clearTimeout(timmer);

if (d !== 0) {

timmer = setTimeout(() => {

this.handleHide();

timmer = null;

}, d);

}

},

handleHide () {

this.setData({

...default\_data

});

}

}

});

<!--index.wxml-->

<view class="container">

<scroll-view class='content'

scroll-y

bindscrolltolower="loadMore"

scroll-with-animation="true"

scroll-top="{{topPosition}}"

bindscroll="scrollPosition">

<w-swiper images="{{banners}}"/>

<!-- <txv-video vid="h3019qbsbuc" playerid="txv1"></txv-video> -->

<view class="section">

<!-- <image src="../../images/generic\_countdown.png" mode="widthFix" style="width:100%;"></image> -->

<video id="myVideo" src="{{activityLive}}" class="video\_src" controls></video>

</view>

<navigator url="../../components/w-ad/w-ad">

<image src="../../images/Light\_Red\_Pot.jpg" mode="widthFix" style="width:100%;"></image>

</navigator>

<w-recommend recommends="{{recommends}}" />

<i-tabs current="{{current}}" color="#f759ab" bindchange="switchPage">

<i-tab key="1" title="投票"></i-tab>

<i-tab key="2" title="公告栏"></i-tab>

</i-tabs>

<view wx:if="{{tab1}}" class='tab1'>

<w-vote

class="src"

votesList="{{votes}}"

recordNickname="{{nickName}}"

openid="{{getOpenID}}"

bind:eventListener="handleEventListener"

></w-vote>

<!-- <view class="container">

<chatroom

style="width: 100%; height: 500rpx"

envId="{{chatRoomEnvId}}"

collection="{{chatRoomCollection}}"

groupId="{{chatRoomGroupId}}"

userInfo="{{userInfo}}"

onGetUserInfo="{{onGetUserInfo}}"

getOpenID="{{getOpenID}}"

></chatroom>

</view> -->

</view>

<view wx:if="{{tab2}}" class='tab2'>

<w-note

class="src"

notesList="{{notes}}"

></w-note>

</view>

</scroll-view>

</view>

//index.js

var time = require('../../utils/time.js');

var util = require('../../utils/common.js');

const app = getApp()

export const BACK\_TOP\_POSITION = 1000;

Page({

data: {

// i-tab切换页面相关

current: '1',

tab1: true,

tab2: false,

// 主页所需的各种数据

activityId: '',

activityName: '',

activityProfile: '',

activityLive: '',

banners: ,

votes: ,

notes: ,

recommends: ,

nickName: null,

getOpenID: null,

userInfo: {},

hasUserInfo: false,

canIUse: wx.canIUse('button.open-type.getUserInfo')

// 聊天室相关

// topPosition: 0,

// avatarUrl: './user-unlogin.png',

// userInfo: null,

// logged: false,

// takeSession: false,

// requestResult: '',

// chatRoomEnvId: 'release-f8415a',

// chatRoomCollection: 'chatroom',

// chatRoomGroupId: 'demo',

// chatRoomGroupName: '聊天室',

// 在聊天室组件中需要用到的方法

// onGetUserInfo: null,

},

// 初始化

onLoad: function (options) {

// 发送网络请求

this.getLogin()

this.getInfo()

this.getActivity()

// 获得用户手机信息

wx.getSystemInfo({

success: res => {

// console.log('system info', res)

if (res.safeArea) {

const { top, bottom } = res.safeArea

this.setData({

containerStyle: `padding-top: ${(/ios/i.test(res.system) ? 10 : 20) + top}px; padding-bottom: ${20 + res.windowHeight - bottom}px`,

})

}

},

})

},

// 初始化结束，onShow用来刷新当前页面的

onShow: function () {

const pages = getCurrentPages()

const currentPage = pages

this.onLoad(currentPage.options);

},

// 预加载视频&直播

onReady: function (res) {

this.videoContext = wx.createVideoContext('myVideo')

},

// 下拉刷新，测试使用

onPullDownRefresh: function (){

this.onLoad();

wx.stopPullDownRefresh();

wx.hideLoading();

},

// 监听投票组件是否被点击，检测到被点击的时候执行onShow方法刷新页面

handleEventListener() {

this.onShow();

console.log('更新票数')

},

// 分享页面时执行的操作

onShareAppMessage: function (ops) {

var shareObj = {

title: this.data.activityName,

path: 'pages/index/index', // 路径，传递参数到指定页面。

imageUrl: '../../assets/images/tabbar/pic\_share.jpg', // 分享的封面图

}

return shareObj;

},

// tab分页栏的控制方法

switchPage({ detail }) {

var index = detail.key

this.setData({

current: detail.key

});

if (index == 1) {

this.setData({

tab1: true,

tab2: false

})

} else if (index == 2) {

this.setData({

tab1: false,

tab2: true

})

}

},

getLogin(){

let that = this

if (app.checkCache()) {

wx.redirectTo({

url: 'pages/index/index',

})

} else {

wx.showToast({

title: '当前账户未登陆，请先登陆',

icon: 'none',

duration: 3000

})

app.login()

}

that.setData({

// onGetUserInfo: this.onGetUserInfo,

getOpenID: wx.getStorageSync("openid"),

})

},

getInfo(){

if (app.globalData.userInfo) {

this.setData({

userInfo: app.globalData.userInfo,

hasUserInfo: true

})

} else if (this.data.canIUse) {

// 由于 getUserInfo 是网络请求，可能会在 Page.onLoad 之后才返回

// 所以此处加入 callback 以防止这种情况

app.userInfoReadyCallback = res => {

this.setData({

userInfo: res.userInfo,

hasUserInfo: true

})

}

} else {

// 在没有 open-type=getUserInfo 版本的兼容处理

wx.getUserInfo({

success: res => {

app.globalData.userInfo = res.userInfo

this.setData({

userInfo: res.userInfo,

hasUserInfo: true

})

}

})

}

},

getActivity(){

let that = this

var data = {

a: '123',

}

util.getAPI("POST", data, 'activity/find\_active').then((res) => {

if (res.data.code === 200) {

app.globalData.activityId = res.data.data.id

app.globalData.activityName = res.data.data.activityName

app.globalData.activityProfile = res.data.data.activityProfile

that.setData({

activityId: res.data.data.id,

activityName: res.data.data.activityName,

activityLive: res.data.data.activityLive,

activityProfile: res.data.data.activityProfile,

})

}

wx.setNavigationBarTitle({

title: res.data.data.activityName,

})

}).then(()=>{

that.getNote()

}).then(() => {

that.getBanners()

}).then(() => {

that.getVote()

}).then(() => {

that.getRecommends()

})

},

//云端根据时间返回当前活动详情

getNote() {

let that = this;

var activityId = this.data.activityId

var data = {

id: activityId,

}

util.getAPI("GET", data, 'note/find\_active').then((res) => {

if (res.data.code === 200) {

let dataList = res.data.data;

Object.keys(dataList).forEach(key => (dataList.noteDisplayTime = time.js\_date\_time(dataList.noteDisplayTime)))

that.setData({

notes: dataList

})

}

})

},

// 获取轮播图

getBanners() {

let that = this;

let imgArr = ;

var activityId = this.data.activityId

var data = {

id: activityId,

}

util.getAPI("GET", data, 'swiper/find\_active').then((res) => {

if (res.data.code === 200) {

let dataList = res.data.data;

for (let i = 0; i < dataList.length; i++) {

imgArr.push(app.globalData.callbackUrl + dataList.swiperPic)

}

that.setData({

banners: imgArr

})

}

})

},

// 获取投票对象

getVote(){

let that = this;

var activityId = this.data.activityId

var data = {

id: activityId,

}

util.getAPI("GET", data, 'group/find\_active').then((res) => {

if (res.data.code === 200) {

let dataList = res.data.data;

Object.keys(dataList).forEach(key => (dataList.groupPicHd = app.globalData.mpCallbackUrl + dataList.groupPicHd))

that.setData({

votes: dataList

})

}

})

},

// 获取参与单位

getRecommends(){

let that = this;

var activityId = this.data.activityId

var data = {

id: activityId,

}

util.getAPI("GET", data, 'merchant/find\_active').then((res) => {

if (res.data.code === 200) {

let dataList = res.data.data;

Object.keys(dataList).forEach(key => (dataList.merchantLogo = app.globalData.callbackUrl + dataList.merchantLogo))

that.setData({

recommends: dataList

})

}

})

},

// 边滑动变加载，防爬虫防刷票防卡死

scrollPosition(e) {

// 1.获取滚动的顶部

const position = e.detail.scrollTop;

// 2.设置是否显示

this.setData({

showBackTop: position > BACK\_TOP\_POSITION,

})

wx.createSelectorQuery().select('.src').boundingClientRect((rect) => {

const show = rect.top > 0

this.setData({

showTabControl: !show

})

}).exec()

}

})

Component({

externalClasses: ,

properties: {

// default, primary, ghost, info, success, warning, error

type: {

type: String,

value: '',

},

inline: {

type: Boolean,

value: false

},

// default, large, small

size: {

type: String,

value: '',

},

// circle, square

shape: {

type: String,

value: 'square'

},

disabled: {

type: Boolean,

value: false,

},

loading: {

type: Boolean,

value: false,

},

long: {

type: Boolean,

value: false

},

openType: String,

appParameter: String,

hoverStopPropagation: Boolean,

hoverStartTime: {

type: Number,

value: 20

},

hoverStayTime: {

type: Number,

value: 70

},

lang: {

type: String,

value: 'en'

},

sessionFrom: {

type: String,

value: ''

},

sendMessageTitle: String,

sendMessagePath: String,

sendMessageImg: String,

showMessageCard: Boolean

},

methods: {

handleTap () {

if (this.data.disabled) return false;

this.triggerEvent('click');

},

bindgetuserinfo({ detail = {} } = {}) {

this.triggerEvent('getuserinfo', detail);

},

bindcontact({ detail = {} } = {}) {

this.triggerEvent('contact', detail);

},

bindgetphonenumber({ detail = {} } = {}) {

this.triggerEvent('getphonenumber', detail);

},

binderror({ detail = {} } = {}) {

this.triggerEvent('error', detail);

}

}

});

<view class="i-class i-card {{ full ? 'i-card-full' : '' }}">

<view class="i-class i-card-header">

<view class="i-card-header-content">

<image class="i-card-header-thumb" src="{{ thumb }}" mode="aspectFit" wx:if="{{ thumb }}" />

{{ title }}

</view>

<view class="i-card-header-extra" wx:if="{{ extra }}">{{ extra }}</view>

</view>

<view class="i-class i-card-body"><slot name="content"></slot></view>

<view class="i-class i-card-footer"><slot name="footer"></slot></view>

</view>

.i-card{margin:0 16px;font-size:14px;overflow:hidden;position:relative;background:#fff;border:1rpx solid #dddee1;border-radius:5px}.i-card-full{margin:0;border-left:none;border-right:none;border-radius:0}.i-card-header{display:flex;padding:6px 16px;align-items:center}.i-card-header-content{flex:1;text-align:left}.i-card-header-thumb{display:inline-block;width:64px;height:64px;position:relative;margin-left:auto;margin-right:auto;overflow:hidden;background-size:cover;vertical-align:middle}.i-card-header-title{display:inline-block;vertical-align:middle;font-size:14px;color:#1c2438}.i-card-header-extra{flex:1;text-align:right;font-size:14px;color:#80848f}.i-card-body{position:relative;padding:6px 16px;color:#495060;font-size:14px}.i-card-body::before{content:'';position:absolute;top:0;left:0;width:200%;height:200%;transform:scale(.5);transform-origin:0 0;pointer-events:none;box-sizing:border-box;border:0 solid #e9eaec;border-top-width:1px}.i-card-footer{position:relative;padding:6px 16px;color:#80848f;font-size:12px}

Component({

externalClasses: ,

options: {

multipleSlots: true

},

properties: {

full: {

type: Boolean,

value: false

},

thumb: {

type: String,

value: ''

},

title: {

type: String,

value: ''

},

extra: {

type: String,

value: ''

}

}

});

<view bindtap="handleTap" class="i-class i-cell {{ isLastCell ? 'i-cell-last' : '' }} {{ isLink ? 'i-cell-access' : '' }}">

<view class="i-cell-icon">

<slot name="icon"></slot>

</view>

<view class="i-cell-bd">

<view wx:if="{{ title }}" class="i-cell-text">{{ title }}</view>

<view wx:if="{{ label }}" class="i-cell-desc">{{ label }}</view>

<slot></slot>

</view>

<view catchtap="navigateTo" class="i-cell-ft">

<block wx:if="{{value}}">{{ value }}</block>

<block wx:else>

<slot name="footer"></slot>

</block>

</view>

</view>.i-cell{position:relative;padding:12px 15px;display:flex;background:#fff;align-items:center;line-height:1.4;font-size:14px;overflow:hidden}.i-cell::after{content:'';position:absolute;top:0;left:0;width:200%;height:200%;transform:scale(.5);transform-origin:0 0;pointer-events:none;box-sizing:border-box;border:0 solid #e9eaec;border-bottom-width:1px;left:15px;right:0}.i-cell-last::after{display:none}.i-cell-icon{margin-right:5px}.i-cell-icon:empty{display:none}.i-cell-bd{flex:1}.i-cell-text{line-height:24px;font-size:14px}.i-cell-desc{line-height:1.2;font-size:12px;color:#80848f}.i-cell-ft{position:relative;text-align:right;color:#495060}.i-cell-access .i-cell-ft{padding-right:13px}.i-cell-access .i-cell-ft::after{content:" ";display:inline-block;width:6px;height:6px;position:absolute;top:50%;right:2px;border-width:2px 2px 0 0;border-color:#dddee1;border-style:solid;transform:translateY(-50%) matrix(.71,.71,-.71,.71,0,0)}

const warn = (msg, getValue) => {

console.warn(msg);

console.log('接受到的值为：', getValue);

};

Component({

externalClasses: ,

options: {

multipleSlots: true

},

relations: {

'../cell-group/index': {

type: 'parent'

}

},

properties: {

// 左侧标题

title: {

type: String

},

// 标题下方的描述信息

label: {

type: String

},

// 右侧内容

value: {

type: String

},

// 只有点击 footer 区域才触发 tab 事件

onlyTapFooter: {

type: Boolean

},

// 是否展示右侧箭头并开启尝试以 url 跳转

isLink: {

type: null,

value: ''

},

// 链接类型，可选值为 navigateTo，redirectTo，switchTab，reLaunch

linkType: {

type: String,

value: 'navigateTo'

},

url: {

type: String,

value: ''

}

},

data: {

isLastCell: true

},

methods: {

navigateTo () {

const { url } = this.data;

const type = typeof this.data.isLink;

this.triggerEvent('click', {});

if (!this.data.isLink || !url || url === 'true' || url === 'false') return;

if (type !== 'boolean' && type !== 'string') {

warn('isLink 属性值必须是一个字符串或布尔值', this.data.isLink);

return;

}

if (.indexOf(this.data.linkType) === -1) {

warn('linkType 属性可选值为 navigateTo，redirectTo，switchTab，reLaunch', this.data.linkType);

return;

}

wx.call(wx, {url});

},

handleTap () {

if (!this.data.onlyTapFooter) {

this.navigateTo();

}

},

updateIsLastCell (isLastCell) {

this.setData({ isLastCell });

}

}

});

<view class="i-class i-cell-group">

<slot></slot>

</view>.i-checkbox-cell::after{display:block}.i-checkbox-checkbox-left{float:left}.i-checkbox-checkbox-right{float:right}.i-checkbox-radio{vertical-align:middle}.i-checkbox-title{display:inline-block;vertical-align:middle}

Component({

externalClasses: ,

relations: {

'../cell/index': {

type: 'child',

linked () {

this.\_updateIsLastCell();

},

linkChanged () {

this.\_updateIsLastCell();

},

unlinked () {

this.\_updateIsLastCell();

}

}

},

methods: {

\_updateIsLastCell() {

let cells = this.getRelationNodes('../cell/index');

const len = cells.length;

if (len > 0) {

let lastIndex = len - 1;

cells.forEach((cell, index) => {

cell.updateIsLastCell(index === lastIndex);

});

}

}

}

});

<view class="i-class i-checkbox" catchtap="checkboxChange">

<i-cell i-class="i-checkbox-cell">

<label>

<radio value="{{value}}" checked="{{checked}}" color="{{checked?color:''}}" disabled="{{disabled}}" class="i-checkbox-radio {{positionCls}}" />

<view class="i-checkbox-title">{{value}}</view>

</label>

</i-cell>

</view>

Component({

externalClasses: ,

options: {

multipleSlots: true

},

properties: {

visible: {

type: Boolean,

value: false

},

maskClosable: {

type: Boolean,

value: true

},

showCancel: {

type: Boolean,

value: false

},

cancelText: {

type: String,

value: '取消'

},

actions: {

type: Array,

value:

}

},

methods: {

handleClickMask () {

if (!this.data.maskClosable) return;

this.handleClickCancel();

},

handleClickItem ({ currentTarget = {} }) {

const dataset = currentTarget.dataset || {};

const { index } = dataset;

this.triggerEvent('click', { index });

},

handleClickCancel () {

this.triggerEvent('cancel');

}

}

});

<view class="i-as-mask i-class-mask {{ visible ? 'i-as-mask-show' : '' }}" bindtap="handleClickMask"></view>

<view class="i-class i-as {{ visible ? 'i-as-show' : '' }}">

<view class="i-as-header i-class-header"><slot name="header"></slot></view>

<view class="i-as-actions">

<view class="i-as-action-item" wx:for="{{ actions }}" wx:key="{{ item.name }}">

<i-button

bind:click="handleClickItem"

data-index="{{ index }}"

open-type="{{ item.openType }}"

type="ghost"

size="large"

long

>

<view class="i-as-btn-loading" wx:if="{{ item.loading }}"></view>

<i-icon wx:if="{{ item.icon }}" type="{{ item.icon }}" i-class="i-as-btn-icon"></i-icon>

<view class="i-as-btn-text" style="{{ item.color ? 'color: ' + item.color : '' }}">{{ item.name }}</view>

</i-button>

</view>

</view>

<view class="i-as-cancel" wx:if="{{ showCancel }}">

<i-button i-class="i-as-cancel-btn" type="ghost" size="large" long="true" bind:click="handleClickCancel">{{ cancelText }}</i-button>

</view>

</view>

.i-as{position:fixed;width:100%;box-sizing:border-box;left:0;right:0;bottom:0;background:#f7f7f7;transform:translate3d(0,100%,0);transform-origin:center;transition:all .2s ease-in-out;z-index:900;visibility:hidden}.i-as-show{transform:translate3d(0,0,0);visibility:visible}.i-as-mask{position:fixed;top:0;left:0;right:0;bottom:0;background:rgba(0,0,0,.7);z-index:900;transition:all .2s ease-in-out;opacity:0;visibility:hidden}.i-as-mask-show{opacity:1;visibility:visible}.i-as-action-item{position:relative}.i-as-action-item::after{content:'';position:absolute;top:0;left:0;width:200%;height:200%;transform:scale(.5);transform-origin:0 0;pointer-events:none;box-sizing:border-box;border:0 solid #e9eaec;border-bottom-width:1px}.i-as-header{background:#fff;text-align:center;position:relative;font-size:12px;color:#80848f}.i-as-header::after{content:'';position:absolute;top:0;left:0;width:200%;height:200%;transform:scale(.5);transform-origin:0 0;pointer-events:none;box-sizing:border-box;border:0 solid #e9eaec;border-bottom-width:1px}.i-as-cancel{margin-top:6px}.i-as-btn-loading{display:inline-block;vertical-align:middle;margin-right:10px;width:12px;height:12px;background:0 0;border-radius:50%;border:2px solid #e5e5e5;border-color:#666 #e5e5e5 #e5e5e5 #e5e5e5;animation:btn-spin .6s linear;animation-iteration-count:infinite}.i-as-btn-text{display:inline-block;vertical-align:middle}.i-as-btn-icon{font-size:14px!important;margin-right:4px}@keyframes btn-spin{0%{transform:rotate(0)}100%{transform:rotate(360deg)}}

// miniprogram/pages/promotion/promotion.js

var util = require('../../utils/common.js');

const app = getApp()

Page({

/\*\*

\* 页面的初始数据

\*/

data: {

merchant:,

},

/\*\*

\* 生命周期函数--监听页面加载

\*/

onLoad: function (options) {

this.getMerchant();

},

getMerchant() {

let that = this;

var activityId = app.globalData.activityId

var data = {

id: activityId,

}

util.getAPI("GET", data, 'merchant/find\_active').then((res) => {

if (res.data.code === 200) {

let dataList = res.data.data;

Object.keys(dataList).forEach(key => (dataList.merchantLogo = app.globalData.callbackUrl + dataList.merchantLogo))

that.setData({

merchant: dataList

})

}

})

},

/\*\*

\* 生命周期函数--监听页面初次渲染完成

\*/

onReady: function () {

},

/\*\*

\* 生命周期函数--监听页面显示

\*/

onShow: function () {

},

/\*\*

\* 生命周期函数--监听页面隐藏

\*/

onHide: function () {

},

/\*\*

\* 生命周期函数--监听页面卸载

\*/

onUnload: function () {

},

/\*\*

\* 页面相关事件处理函数--监听用户下拉动作

\*/

onPullDownRefresh: function () {

},

/\*\*

\* 页面上拉触底事件的处理函数

\*/

onReachBottom: function () {

},

/\*\*

\* 用户点击右上角分享

\*/

})<!--miniprogram/pages/promotion/promotion.wxml-->

<view class="container">

<view class="merchant\_list">

<view class="each\_merchant" wx:for="{{merchant}}" wx:key="index">

<view class="pic">

<image src="{{item.merchantLogo}}" mode="widthFix"></image>

</view>

<view class="merchant\_info">

<view class="merchant\_name">

<text>

{{item.merchantName}}

</text>

</view>

<view class="merchant\_introduction">

<text>

{{item.merchantInfo}}

</text>

</view>

</view>

</view>

</view>

</view>

/\* miniprogram/pages/promotion/promotion.wxss \*/

.container {

height: 100%;

top: 0;

bottom: 0;

left: 0;

right: 0;

padding-top: 0rpx;

padding-bottom: 0rpx;

background-image: url("https://migotimes.com/file/mp/bg.png");

background-size: cover;

-webkit-background-size: cover;

-o-background-size: cover;

}

.each\_merchant{

display: -webkit-box;

display: -webkit-flex;

display: flex;

-webkit-box-align: center;

-webkit-align-items: center;

align-items: center;

border-bottom: 1px dashed rgb(85, 85, 85);

}

.pic{

margin-right: .8em;

width: 25%;

/\* height: 60px; \*/

/\* line-height: 60px; \*/

text-align: center;

}

.pic image{

width: 100%;

/\* height: 100%; \*/

vertical-align: bottom;

}

.merchant\_name{

font-weight: 400;

font-size: 17px;

width: auto;

overflow: hidden;

text-overflow: ellipsis;

white-space: nowrap;

word-wrap: break-word;

word-break: break-all;

color:rgb(58, 58, 58);

}

.merchant\_info{

padding-bottom: 5%;

-webkit-box-flex: 1;

-webkit-flex: 1;

flex: 1;

min-width: 0;

}

.merchant\_introduction{

/\* color: #999999; \*/

font-size: 13px;

line-height: 1.2;

overflow: hidden;

text-overflow: ellipsis;

-webkit-box-orient: vertical;

-webkit-line-clamp: 4;

color:rgb(58, 58, 58);

}

.merchant\_list{

/\* margin-top: 5rpx; \*/

background-color: rgba(158, 232, 255, 0.76);

padding: 0rpx 50rpx 50rpx 50rpx;

clear: both;

overflow: hidden;

position: relative;

}

.for\_details{

float: right;

}

.title{

padding: 0 30rpx 50rpx 30rpx;

}

// miniprogram/pages/propaganda/propaganda.js

var time = require('../../utils/time.js');

var util = require('../../utils/common.js');

const app = getApp()

Page({

/\*\*

\* 页面的初始数据

\*/

data: {

notesList:,

current: '1',

tab1: true,

tab2: false,

activityProfile: '',

notes: ,

},

/\*\*

\* 生命周期函数--监听页面加载

\*/

onLoad: function (options) {

this.getAct();

this.getNote();

},

//发请求给云函数，云端根据时间返回当前活动详情（不同的时间返回的值是不一样的，由人工控制）

getAct() {

let that = this;

that.setData({

activityProfile: app.globalData.activityProfile

})

wx.setNavigationBarTitle({

title: app.globalData.activityName,

})

},

getNote() {

let that = this;

var activityId = app.globalData.activityId

var data = {

id: activityId,

}

util.getAPI("GET", data, 'note/find\_active').then((res) => {

if (res.data.code === 200) {

let dataList = res.data.data;

Object.keys(dataList).forEach(key => (dataList.noteDisplayTime = time.js\_date\_time(dataList.noteDisplayTime)))

that.setData({

notes: dataList

})

}

})

},

switchPage({ detail }) {

var index = detail.key

// console.log(detail.key)

this.setData({

current: detail.key

});

if (index == 1) {

this.setData({

tab1: true,

tab2: false

})

} else if (index == 2) {

this.setData({

tab1: false,

tab2: true

})

}

},

/\*\*

\* 生命周期函数--监听页面初次渲染完成

\*/

onReady: function () {

},

/\*\*

\* 生命周期函数--监听页面显示

\*/

onShow: function () {

},

/\*\*

\* 生命周期函数--监听页面隐藏

\*/

onHide: function () {

},

/\*\*

\* 生命周期函数--监听页面卸载

\*/

onUnload: function () {

},

/\*\*

\* 页面相关事件处理函数--监听用户下拉动作

\*/

onPullDownRefresh: function () {

},

/\*\*

\* 页面上拉触底事件的处理函数

\*/

onReachBottom: function () {

},

/\*\*

\* 用户点击右上角分享

\*/

onShareAppMessage: function (ops) {

var shareObj = {

title: app.globalData.activityName,

path: 'pages/index/index', // 路径，传递参数到指定页面。

imageUrl: '../../assets/images/tabbar/pic\_share.jpg', // 分享的封面图

}

}

})<!--miniprogram/pages/propaganda/propaganda.wxml-->

<!-- 科普列表 -->

<view class="container">

<i-tabs class="tab" current="{{current}}" color="#f759ab" bindchange="switchPage">

<i-tab key="1" title="活动简介"></i-tab>

<i-tab key="2" title="演练流程"></i-tab>

</i-tabs>

<view wx:if="{{tab1}}" class='tab1'>

<view class="note\_wrap">

<!-- 科普组件 -->

<view class="note\_item">

<!-- 科普标题 -->

<view class="note\_title">

<text>

活动宗旨

</text>

</view>

<!-- 科普内容 -->

<view class="note\_content">

<text>

{{activityProfile}}

</text>

</view>

</view>

</view>

</view>

<view wx:if="{{tab2}}" class='tab2'>

<w-note

class="src"

notesList="{{notes}}"

></w-note>

</view>

</view>/\* miniprogram/pages/propaganda/propaganda.wxss \*/

.note\_wrap{

padding: 70rpx;

}

.note\_item{

padding: 30rpx;

/\* width: 100%; \*/

background-color: rgba(163, 225, 255, 0.8);

border: 1px solid rgb(255, 167, 167);

border-radius:5%;

color: rgb(66, 66, 66);

}

.note\_title{

text-align: center;

padding: 0rpx 0 0 0;

font-size: 120%;

margin-top: -12rpx;

}

.note\_content{

padding: 0rpx 0rpx 40rpx 20rpx;

line-height: 50rpx;

}

.container {

height: 100%;

top: 0;

bottom: 0;

left: 0;

right: 0;

padding-top: 0rpx;

padding-bottom: 0rpx;

}

.tab{

width: 100%;

}

function js\_date\_time(unixtime) {

var dateTime = new Date(parseInt(unixtime))

var year = dateTime.getFullYear();

var month = dateTime.getMonth() + 1;

var \_month = month >= 10 ? month : "0" + month;

var day = dateTime.getDate();

var \_day = day >= 10 ? day : "0" + day;

var hour = dateTime.getHours();

var \_hour = hour >= 10 ? hour : "0" + hour;

var minute = dateTime.getMinutes();

var \_minute = minute >= 10 ? minute : "0" + minute;

var second = dateTime.getSeconds();

var \_second = second >= 10 ? second : "0" + second;

var now = new Date();

var now\_new = Date.parse(now.toDateString()); //typescript转换写法

var milliseconds = now\_new - dateTime;

var timeSpanStr = year + '-' + \_month + '-' + \_day + ' ' + \_hour + ':' + \_minute + ':' + \_second;

return timeSpanStr;

}

module.exports = {

js\_date\_time: js\_date\_time

}

// miniprogram/pages/vote/vote.js

var util = require('../../utils/common.js');

const app = getApp()

Page({

/\*\*

\* 页面的初始数据

\*/

data: {

rankList: ,

nickName: null,

votes:

},

/\*\*

\* 生命周期函数--监听页面加载

\*/

onLoad: function (options) {

this.getActivity();

this.getRank();

this.getVote();

// 获取用户信息

wx.getSetting({

success: res => {

if (res.authSetting) {

// 已经授权，可以直接调用 getUserInfo 获取头像昵称，不会弹框

wx.getUserInfo({

success: res => {

// console.log(res.userInfo)

this.setData({

nickName: res.userInfo.nickName,

})

}

})

}

}

})

},

onShareAppMessage: function (ops) {

var shareObj = {

title: '省三防办·肇庆市灾难应急演练',

path: 'pages/index/index', // 路径，传递参数到指定页面。

imageUrl: '../../assets/images/tabbar/pic\_share.jpg', // 分享的封面图

}

return shareObj;

},

getActivity() {

let that = this

wx.setNavigationBarTitle({

title: app.globalData.activityName,

})

},

getRank() {

let that = this;

var activityId = app.globalData.activityId

var data = {

id: activityId,

}

util.getAPI("GET", data, 'group/find\_rank').then((res) => {

if (res.data.code === 200) {

let dataList = res.data.data;

// console.log(dataList)

Object.keys(dataList).forEach(key => (dataList.groupPicHd = app.globalData.mpCallbackUrl + dataList.groupPicHd))

that.setData({

rankList: dataList

})

}

})

},

getVote() {

let that = this;

var activityId = app.globalData.activityId

var data = {

id: activityId,

}

util.getAPI("GET", data, 'group/find\_active').then((res) => {

if (res.data.code === 200) {

let dataList = res.data.data;

// console.log(dataList)

Object.keys(dataList).forEach(key => (dataList.groupPicHd = app.globalData.mpCallbackUrl + dataList.groupPicHd))

that.setData({

votes: dataList

})

}

})

},

handleEventListener() {

this.onShow();

console.log('更新票数')

},

/\*\*

\* 生命周期函数--监听页面初次渲染完成

\*/

onReady: function () {

},

/\*\*

\* 生命周期函数--监听页面显示

\*/

onShow: function () {

const pages = getCurrentPages()

const currentPage = pages

this.onLoad(currentPage.options);

},

/\*\*

\* 生命周期函数--监听页面隐藏

\*/

onHide: function () {

},

/\*\*

\* 生命周期函数--监听页面卸载

\*/

onUnload: function () {

},

/\*\*

\* 页面相关事件处理函数--监听用户下拉动作

\*/

onPullDownRefresh: function () {

},

/\*\*

\* 页面上拉触底事件的处理函数

\*/

onReachBottom: function () {

},

/\*\*

\* 用户点击右上角分享

\*/

})<!--miniprogram/pages/vote/vote.wxml-->

<view class="all\_votes">

<view class="presentation">

人气榜

</view>

<view class="rankings" wx:for="{{rankList}}" wx:key="index">

<!-- 图片容器 -->

<navigator url="../../pages/vote\_detail/vote\_detail?id={{item.id}}" class="rank\_img">

<image src="{{item.groupPicHd}}" mode="widthFix"></image>

</navigator>

<!-- 描述 -->

<view class="rank\_info">

<!-- 标题 -->

<view class="rank\_title">

{{item.groupName}}

</view>

<!-- 票数相关 -->

<view class="rank\_vote">

<!-- 分隔图标 -->

<view class="vote\_icon\_color">

<text class="fa fa-flag"></text>

<!-- 票数 -->

{{item.groupVotes}}

</view>

</view>

</view>

<view class="model">

<image wx:if="{{index==0}}" mode="widthFix" src="../../assets/images/tabbar/gold.png"></image>

<image wx:elif="{{index==1}}" mode="widthFix" src="../../assets/images/tabbar/silver.png"></image>

<image wx:elif="{{index==2}}" mode="widthFix" src="../../assets/images/tabbar/copper.png"></image>

</view>

</view>

</view>

<view class="vote">

<view class="presentation2">

你最想看的演练项目

</view>

<w-vote

class="src"

votesList="{{votes}}"

nickName="{{nickName}}"

bind:eventListener="handleEventListener"

></w-vote>

</view>/\* miniprogram/pages/vote/vote.wxss \*/

.all\_votes{

align-content: stretch;

padding: 30rpx;

align-items: center;

}

.rankings{

background-color: rgba(163, 225, 255, 0.3);

/\* position: relative; \*/

display: block;

overflow: hidden;

align-items: center;

margin: 30rpx;

width: 90%;

border: 1rpx solid rgb(255, 188, 219);

border-radius: 20rpx 20rpx 20rpx 20rpx;

}

.presentation{

margin: 60rpx;

background-color: rgba(163, 225, 255, 0.8);

border: 1rpx solid rgb(255, 188, 219);

border-radius: 20rpx 20rpx 20rpx 20rpx;

text-align: center;

padding: 20rpx;

font-size: 120%;

color: rgb(66, 66, 66);

}

.presentation2{

margin: 60rpx;

margin-bottom: 12rpx;

background-color: rgba(163, 225, 255, 0.8);

border: 1rpx solid rgb(255, 188, 219);

border-radius: 20rpx 20rpx 20rpx 20rpx;

text-align: center;

padding: 20rpx;

font-size: 120%;

color: rgb(66, 66, 66);

}

.rank\_title{

padding-top: 3px;

font-size: 70%;

color: rgb(75, 75, 75);

}

.rank\_img{

}

.rank\_info{

display: block;

float: left;

padding: 3% 0 3% 0;

}

.rank\_vote{

/\* position: absolute; \*/

/\* vertical-align: top; \*/

/\* margin-right: 10rpx; \*/

padding-top: 3px;

margin-left: 1px;

}

.rank\_img image{

display: block;

float: left;

/\* overflow: hidden; \*/

width: 120px;

/\* background-size: cover;

background-position: 50%;

background-repeat: no-repeat; \*/

margin-right: 5%;

}

.model image{

display: block;

float: right;

width: 30px;

padding: 10px;

margin-right: 10px;

}

.vote{

margin-top: -6%;

}

.vote\_icon\_color{

color: rgb(255, 110, 110);

}

// miniprogram/pages/vote\_detail/vote\_detail.js

var util = require('../../utils/common.js');

const app = getApp()

Page({

/\*\*

\* 页面的初始数据

\*/

data: {

group\_profile:"",

pic\_hd:"",

activity\_filters:"",

// vote\_id:""

},

/\*\*

\* 生命周期函数--监听页面加载

\*/

onLoad: function (options) {

// console.log(options)

var vote\_id = options.id;

// console.log(options.id)

this.getCurrentAct(vote\_id);

},

// 根据活动id获取对象详情

//

getCurrentAct(vote\_id){

let that = this;

var data = {

id: vote\_id,

}

util.getAPI("GET", data, 'group/find\_group\_id').then((res) => {

if (res.data.code === 200) {

console.log(res.data.data)

let groupInfo = res.data.data;

groupInfo.groupPicHd = app.globalData.callbackUrl + groupInfo.groupPicHd

that.setData({

groupPicHd: groupInfo.groupPicHd,

groupName: groupInfo.groupName,

groupProfile: groupInfo.groupProfile

})

}

})

},

onShareAppMessage: function (ops) {

var shareObj = {

title: '省三防办·肇庆市灾难应急演练',

path: 'pages/index/index', // 路径，传递参数到指定页面。

imageUrl: '../../assets/images/tabbar/pic\_share.jpg', // 分享的封面图

}

return shareObj;

},

})<!--miniprogram/pages/vote\_detail/vote\_detail.wxml-->

<view class="container">

<image src="{{groupPicHd}}" mode="widthFix" style="width:100%"></image>

<view class='info'>

<view class="title">

{{groupName}}

</view>

<view class="profile">

{{groupProfile}}

</view>

</view>

<view class="guide">

<view class="share\_text">

<text>

点击右侧按钮转发拉票\n投票请返回上一级

</text>

</view>

<button class="share" open-type='share' plain='true' hover-class="share\_active"></button>

</view>

</view>

/\* miniprogram/pages/vote\_detail/vote\_detail.wxss \*/

.container {

height: 100%;

top: 0;

bottom: 0;

left: 0;

right: 0;

padding-top: 0rpx;

padding-bottom: 0rpx;

}

.title{

text-align: center;

padding: 20rpx 0 0 0;

font-size: 120%;

}

.info{

/\* align-items: center; \*/

/\* margin: 100rpx; \*/

padding: 50rpx;

background-color: rgba(163, 225, 255, 0.8);

border-width:3rpx;

border-color:rgb(255, 188, 219);

border-style:solid;

color:rgb(85, 85, 85);

}

.profile{

padding: 30rpx 0rpx 40rpx 20rpx;

}

.share{

background-image: url("https://migotimes.com/file/mp/share\_active.png");

width: 60px;

height: 60px;

float:right;

}

.share\_active{

filter: grayscale(1);

}

.guide{

/\* margin: 100rpx; \*/

padding: 50rpx;

/\* padding: 50rpx; \*/

display: flex;

background-color: rgba(163, 225, 255, 0.8);

position: relative;

align-items: center;

/\* margin: 30rpx; \*/

width: 100%;

border: 1rpx solid rgb(255, 188, 219);

border-radius: 20rpx 20rpx 20rpx 20rpx;

color:rgb(85, 85, 85);

}

.share\_text{

padding: 20rpx 0rpx 60rpx 100rpx;

}

Component({

/\*\*

\* 组件的属性列表

\*/

properties: {

recommends: {

type: Array,

value:

}

},

/\*\*

\* 组件的初始数据

\*/

data: {

},

/\*\*

\* 组件的方法列表

\*/

methods: {

}

})

<!--miniprogram/components/w-ad/w-ad.wxml-->

<official-account></official-account>

<view class="ad\_wrap">

<view class="ad\_item">

请点击上方「查看」按钮，进入「米果传媒」公众号主页获取红包及优惠信息~

</view>

</view>/\* miniprogram/components/w-ad/w-ad.wxss \*/

.ad\_wrap{

background-image: url("https://migotimes.com/file/mp/bg.jpg");

}

.ad\_item{

align-items: center;

margin: 100rpx;

padding: 50rpx;

background-color: rgba(163, 225, 255, 0.8);

border-radius: 10rpx;

/\* border-width:3rpx; \*/

/\* border-color:rgb(255, 188, 219); \*/

/\* border-style:solid; \*/

color:rgb(85, 85, 85);

line-height: 50rpx;

}

// miniprogram/components/w-note/w-note.js

Component({

/\*\*

\* 组件的属性列表

\*/

properties: {

notesList: {

type: Array,

value:

}

},

/\*\*

\* 组件的初始数据

\*/

data: {

},

/\*\*

\* 组件的方法列表

\*/

methods: {

}

})<!--miniprogram/components/w-note/w-note.wxml-->

<!-- 公告列表 -->

<view class="note\_wrap">

<!-- 公告组件 -->

<view class="note\_item" wx:for="{{notesList}}" wx:key="index">

<!-- 公告标题 -->

<view class="note\_title">

{{item.noteName}}

</view>

<!-- 公告时间 -->

<view class="note\_time">

{{item.noteDisplayTime}}

</view>

<!-- 公告内容 -->

<view class="note\_content">

<text>

{{item.noteDetails}}

</text>

</view>

</view>

</view>

/\* miniprogram/components/w-note/w-note.wxss \*/

.note\_wrap{

/\* background: linear-gradient(#86ddff,rgb(130, 235, 229)); \*/

background-image: url("https://migotimes.com/file/mp/bg.png");

box-shadow:0 0 10px 20px rgba(255, 255, 255, 0.438) inset;

background-size: cover;

-webkit-background-size: cover;

-o-background-size: cover;

display: flex;

justify-content: space-between;

align-items: center;

flex-wrap: wrap;

align-content: stretch;

padding: 30rpx;

}

.note\_item{

background-color: rgba(177, 237, 255, 0.4);

border-bottom: 2px solid rgba(169, 210, 255, 0.795);

}

.note\_time{

font-size: 12px;

color: rgba(97, 97, 97, 0.952);

text-align: center;

padding-top:10rpx;

}

.note\_title{

text-align: center;

padding: 50rpx 0 0 0;

font-size: 120%;

color: rgba(75, 75, 75, 0.973);

}

.note\_content{

padding: 0rpx 60rpx 60rpx 60rpx;

color: rgba(75, 75, 75, 0.973);

margin-top: -30rpx;

line-height: 50rpx;

}

// components/w-recommend/w-recommend.js

Component({

/\*\*

\* 组件的属性列表

\*/

properties: {

recommends: {

type: Array,

value:

}

},

/\*\*

\* 组件的初始数据

\*/

data: {

},

/\*\*

\* 组件的方法列表

\*/

methods: {

}

})

<!--components/w-recommend/w-recommend.wxml-->

<navigator url="../../pages/promotion/promotion" open-type='switchTab' class='recommend'>

<view class='recommend-item' wx:for="{{recommends}}" wx:key="index">

<image src="{{item.merchantLogo}}" mode="widthFix" />

<view class="font"><text>{{item.merchantName}}</text></view>

</view>

</navigator>/\* components/w-recommend/w-recommend.wxss \*/

.recommend {

background-color: #2ebdff73;

padding: 20rpx;

display: -webkit-flex;

justify-content: space-between;

flex-wrap: wrap;

/\* margin-top: 20rpx; \*/

align-content: stretch;

/\* border-bottom: 16rpx solid #eee; \*/

}

.recommend-item {

/\* position: relative; \*/

/\* flex: 1; \*/

margin: 20rpx;

width: 25%;

text-align: center;

}

.recommend-item image {

width: 60%;

height: 60%;

border-radius:50%;

}

.recommend-item text {

font-size: 22rpx;

}

.font{

color: rgb(88, 88, 88);

}

// components/w-swiper/w-swiper.js

Component({

/\*\*

\* 组件的属性列表

\*/

properties: {

images: {

type: Array,

value:

}

},

/\*\*

\* 组件的初始数据

\*/

data: {

},

/\*\*

\* 组件的方法列表

\*/

methods: {

}

})

<!--components/w-swiper/w-swiper.wxml-->

<swiper indicator-dots='true'

indicator-active-color='#ff5777'

autoplay='true'

circular='true'

interval='5000'

class='swiper'>

<block wx:for="{{images}}" wx:key="index">

<swiper-item>

<image class="swiper-image" src="{{item}}" mode='widthFix'/>

</swiper-item>

</block>

</swiper>

/\* components/w-swiper/w-swiper.wxss \*/

.swiper {

width: 100%;

height: 250rpx;

}

.swiper-image {

width: 100%;

/\* height: 100%; \*/

}

// miniprogram/components/w-vote/w-vote.js

var util = require('../../utils/common.js');

const app = getApp()

Component({

/\*\*

\* 组件的属性列表

\*/

properties: {

votesList: {

type: Array,

value:

},

openid: String,

id: String

},

/\*\*

\* 组件的初始数据

\*/

data: {

openid: '',

votesList: ,

},

/\*\*

\* 组件的方法列表

\*/

methods: {

addVote: function(e){

let that = this;

var data = {

groupId: e.currentTarget.id,

recordOpenid: this.data.openid

}

// console.log(data)

util.getAPI("POST", data, 'record/add').then((res) => {

// console.log(res)

if (res.data.data === "success") {

wx.showToast({

title: '投票成功',

icon: 'success',

duration: 2000

})

}

else{

wx.showToast({

title: '您已投票',

icon: 'none',

duration: 2000

})

}

})

this.triggerEvent('eventListener')

},

}

})<!--miniprogram/components/w-vote/w-vote.wxml-->

<!-- 投票对象列表 -->

<view class="vote\_wrap">

<!-- 投票组件 -->

<view class="vote\_item" wx:for="{{votesList}}" wx:key="index">

<!-- 图片容器 -->

<navigator url="../../pages/vote\_detail/vote\_detail?id={{item.id}}">

<view class="vote\_img">

<image src="{{item.groupPicHd}}" mode="widthFix"></image>

</view>

<!-- 描述 -->

<view class="vote\_info">

<!-- 标题 -->

<view class="vote\_title">

{{item.groupName}}

</view>

<!-- 分隔图标 -->

<view class="vote\_icon\_color">

<text class="fa fa-flag"></text>

</view>

<!-- 票数 -->

<view class="vote\_count">

{{item.groupVotes}}

</view>

</view>

</navigator>

<!-- 投票按钮 -->

<view class="vote\_btn">

<button bindtap="addVote" class='btn' hover-class='hover-btn' id="{{item.id}}" openid="{{openid}}" size="mini">投票</button>

</view>

</view>

</view>

/\* miniprogram/components/w-vote/w-vote.wxss \*/

@import "../../style/fontawesome.wxss";

.vote\_wrap{

background-image: url("https://migotimes.com/file/mp/bg.png");

background-size: cover;

-webkit-background-size: cover;

-o-background-size: cover;

display: flex;

justify-content: space-between;

align-items: center;

flex-wrap: wrap;

align-content: stretch;

padding: 50rpx;

}

.hover-btn {

/\* background-image:url(../../assets/images/tabbar/vote\_buttom.png); \*/

margin-top: 20rpx;

color: white;

border-radius: 98rpx;

display: flex;

flex-direction: row;

align-items: center;

justify-content: center;

box-shadow:0px 0px 8px rgb(180, 180, 180) inset;

}

.btn {

margin-top: 20rpx;

background-color: rgb(255, 110, 166);

color: white;

border-radius: 98rpx;

display: flex;

flex-direction: row;

align-items: center;

justify-content: center;

}

.vote\_item{

background-color: rgba(163, 225, 255, 0.8);

border-width:3rpx;

border-color:rgb(255, 188, 219);

border-style:solid;

width: 45%;

margin-bottom: 50rpx;

border-radius:20rpx 20rpx 20rpx 20rpx;

}

.vote\_img{

position: relative;

}

.vote\_img image{

width: 100%;

border-radius: 19rpx 19rpx 0 0;

}

.vote\_info{

display: flex;

font-size: 22rpx;

justify-content: space-around;

}

.vote\_btn{

align-items: center;

text-align: center;

}

.vote\_icon\_color{

color: rgb(255, 110, 110)

}

.vote\_title{

color: rgb(88, 88, 88);

}

.vote\_count{

color: rgb(88, 88, 88);

}