// 销毁上一个content页面遗留vueContentObject对象实例

**if** (vueContentObject) getVueObject().$destroy();

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Table \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 定义Table显示相关数据

\* vuePageSize: table每页显示多少条记录

\* loadPageableDataUrl: 分页取数据url

\* tableColumnsName: table每列显示的列头中文名称

\* tableColumnsKey: table每列对应的业务实体类属性名(全选加'selection'项 ,操作加'operation'项)

\* tableButtonsOnEachRow: table中每行需要的业务按钮(以'rowUpdateButton#修改'为例, 其中rowUpdate'Button为修改功能按钮默认按钮名, #号后面标识的'修改'表示想要将按钮名显示为中文'修改')

\*/

[vuePageSize, loadPageableDataUrl, tableColumnsName, tableColumnsKey, tableButtonsOnEachRow] = [

10,

'page',

['','ID','昵称','用户名称','密码','邮箱地址','电话号码','状态','注册日期','最后修改日期','操作'],

['selection','id#sortable','nickname','username#sortable','password','email','phone','status','createdDate','updatedDate','operation'],

['rowUpdateButton#修改','rowDeleteButton#删除']

];

//赋值并格式化table行数据格式

**const** statuses = ['锁定','正常'];

parseValuesOnTableEachRow = obj => ({

id: obj.id,

nickname: obj.nickname,

username: obj.username,

password: obj.password,

email: obj.email,

phone: obj.phone,

status: statuses[obj.status],

createdDate: formatDateTime(obj.createdDate),

updatedDate: formatDateTime(obj.updatedDate)

})

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Forms (queryForm, addForm, updateForm, copyForm, delete functionality) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 创建Vue对象之前定义customVueContentData对象, 使其包含如下数据, 以便后续功能使用:

\* 1. statusDataSelect - queryForm中状态status查询下拉菜单中的options的label和对应的value

\* 2. imgName - 用户照片的名称

\* 3. imgvisibel - 当前的用户照片是否被查看

\* 4. uploadList - 用户上传头像照片列表的容器对象

\*

\* new Vue()生命周期:

\* 1. beforeCreate(创建前)

\* 2. created(创建后)

\* 3. beforeMount(载入前)

\* 4. mounted(载入后)

\* 5. beforeUpdate(更新前)

\* 6. updated(更新后)

\* 7. beforeDestroy(销毁前)

\* 8. destroyed(销毁后)

\*/

vueContentBeforeCreate = () => {

customVueContentData = {

statusDataSelect : statuses.map((val, index, arr) => ({value: index, label: val})),

imgName : '',

imgVisible : **false**,

uploadList : []

}

};

// 设置addForm与 updateForm所对应的 Vue数据模型对象

setFormDataObject({id: -1, username: '', password: '', repassword: '', nickname: '', phone: '', email: '', status: '1', photo: '', photoString: ''});

/\*\*

\* 定义queryForm初始化所需的相关数据

\* queryFormItemName - queryForm中每个item对应显示的查询字段中文名

\* queryFormItemKey - queryForm中每个item对应的业务对象的属性名

\* queryFormItemType - queryForm中每个item的类型(以'select#statusDataSelect'为例,'select'表明当前item应该显示为下拉菜单, 而#号后面的'statusDataSelect'则代表了装载下拉菜单需要加载的下拉选项的对象名)

\*

\* PS: queryFormItemType中以'10<number<20'为例, 所以当前item的类型为数字类型, 且该数字只能取值10与20之间的数字

\*/

[queryFormItemName, queryFormItemKey, queryFormItemType] = [

['ID','昵称','用户名称','密码','邮箱地址','电话号码','状态','注册日期'],

['id','nickname','username','password','email','phone','status','createdDate'],

['string','string','string','string','string','string','select#statusDataSelect','date']

];

// 设置addForm和updateForm中特定表单字段的验证规则

setFormRulesObject({

'username': [{trigger: 'blur',type: 'string', required: **true**, pattern: /^[a-zA-Z\d]\w{4,11}[a-zA-Z\d]$/, message: '用户名称必须为长度6至12位之间以字母、特殊字符(·)或数字字符组成的字符串!'}, {validator: **this**.validateFormRules, trigger: 'blur',unique: 'isUsernameUnique', message: '用户名称已被占用'}],

'password': [{trigger: 'blur',type: 'string', required: **true**, min: 6,max: 16, message: '密码为长度6至12位之间字符串!'}, {validator: **this**.validateFormRules, trigger: 'blur', activateOtherValidation:'repassword'}],

'repassword': [{trigger: 'blur',type: 'string', required: **true**, message:'请输入确认密码'}, {trigger: 'blur', type: 'string', validator: **this**.validateFormRules, equal:'password', message: '两次输入的密码不一致!'}],

'nickname': [{trigger: 'blur',type: 'string', required: **true**, pattern: /^[a-zA-Z0-9·\u4e00-\u9fa5]{2,12}$/, message: '昵称必须为长度2至12位之间以字母、特殊字符(·)、汉字或数组字符组成的字符串!'}, {validator: **this**.validateFormRules, trigger: 'blur',unique: 'isNicknameUnique', message: '昵称已被占用'}],

'phone': [{trigger: 'blur',type: 'string', required: **true**, pattern: /^((1[3,5,8][0-9])|(14[5,7])|(17[0,6,7,8])|(19[7]))\d{8}$/, message: '无效的手机号码格式!'}],

'email': [{trigger: 'blur',type: 'string', required: **true**, pattern: /^([0-9A-Za-z\-\_\.]+)@([0-9a-z]+\.[a-z]{2,3}(\.[a-z]{2})?)$/, message: '无效的电子邮箱格式!'}]

});

//显示更新表单之前: 将确认密码内容填充为用户的密码, 以便用户修改

showUpdateFormBefore = (form, obj) => form.repassword = obj.password;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 上传、查看图片所用功能 开始\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 当上传头像时根据业务需求检查如下内容,当检查过后没有问题,则将该头像放入业务对象属性值中,以待上传:

\* 1. 只能上传1张头像图片

\* 2. 上传头像的格式必须是图片格式 Array [ "jpg", "jpeg", "png" ]

\* 3. 上传头像图片尺寸不能大于2MB(2048 \* 1024)

\*/

vueContentMethods.handleBeforeUpload = **function**(file) {

// 检查: 只能上传1张头像

**const** check = **this**.uploadList.length < 1;

**if** (!check) {

toastError('只能上传1张头像图片');

**return** **false**;

}

// 检查: 上传头像的格式必须是图片格式 Array [ "jpg", "jpeg", "png" ]

**var** uploadComponent = **this**.$refs.upload;

**if** (uploadComponent.format.length) {

**const** fileFormat = file.name.split('.').pop().toLocaleLowerCase();

**const** isImage = uploadComponent.format.some(item => item.toLocaleLowerCase() === fileFormat);

**if** (!isImage) {

handleFormatError(file);

**return** **false**;

}

}

// 检查: 上传头像图片尺寸不能大于2MB(2048 \* 1024)

**if** (uploadComponent.maxSize) {

**if** (file.size > uploadComponent.maxSize \* 1024) {

handleMaxSize(file);

**return** **false**;

}

}

// 读取选择图片数据, 并且为addFrom或updateForm的photo填充图片数据

**var** fileReader = **new** FileReader();

fileReader.onload = **function**(e) {

getVueObject().uploadList.push({'name': file.name, 'url': e.target.result});

**if** (currentAction == actions.add) {

getVueObject().vueAddForm.photoString = e.target.result;

} **else** **if** (currentAction == actions.update) {

getVueObject().vueUpdateForm.photoString = e.target.result;

}

};

fileReader.readAsDataURL(file);

**return** **false**;

}

// 查询头像图片，此方法同样适用于在多个头像中显示某个特定头像

vueContentMethods.handleView = **function**(name) {

**this**.imgName = '';

**let** list = **this**.uploadList;

**for**(**let** i **in** list){

**if**(list[i].name == name) {

**this**.imgName = list[i].url;

**this**.imgVisible = **true**;

**break**;

}

}

}

// 删除头像图片

vueContentMethods.handleRemove = **function**(name) {

**this**.uploadList.map((each, index, array) => {

**if** (each.name == name) {

array.splice(each, 1);

**if** (currentAction == actions.add) {

**this**.vueAddForm.photoString = **null**;

} **else** **if** (currentAction == actions.update) {

**this**.vueUpdateForm.photo = **null**;

**this**.vueUpdateForm.photoString = **null**;

}

}

});

}

// 当头像图片格式不正确,进行提示处理

**function** handleFormatError(file) {

toastError('图片格式不正确(["jpg","jpeg","png"])');

}

// 当头像图片尺寸过大,进行提示处理

**function** handleMaxSize(file) {

toastError('超出文件大小限制,不能超过2MB');

}

// 点击增加按钮,在弹出默认addForm前先将装载头像图片的容器重制制空,以免遗留上一次增加的用户头像

beforeAdd = () => getVueObject().uploadList = [];

// 初始化updateForm后,将装载头像图片的容器重制制空,以免遗留上一次修改的用户头像

// 如果当前要修改的用户业务对象有头像照片,则将用户照片初始化到装载头像的容器内

// 容器中的每个头像图片数据中name保存图片存储后的全路径,url保存其base64格式的图片内容

updateBefore = obj => {

**let** \_this = getVueObject();

\_this.uploadList = [];

**if**(obj.photo && obj.photoString) {

\_this.uploadList.push({'name': obj.photo, 'url': obj.photoString});

\_this.vueUpdateForm.photo = obj.photo;

} **else** {

\_this.vueUpdateForm.photo = **null**;

}

**return** **true**;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 上传、查看图片所用功能 结束\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**var** vueContentObject = **new** Vue(initializeContentOptions());