MES相关前端知识

## map的使用

this.materials = tmpMaterials.map(item => {

return {

materialId: item.materialId,

\_value: false,

materialName: item.materialName, // 添加物料时显示物料名称

quantity: undefined,

productionMode:item.productionMode,

version:'',

versions:[]

}})

## 表格组件

**表头样式**

.el-table\_\_header-wrapper, .el-table\_\_fixed-header-wrapper(固定列表头)

也可使用el-table 自带属性:header-cell-style='getHeaderStyle'设置表头样式，

使用方法，getHeaderStyle() {

return { background: '#f7f7fb', color: '#444' }

},

**表格内容距左边框距离**

// 表格数据距左边距离

.el-table .cell, .el-table th div, .el-table--border td:first-child .cell, .el-table--border th:first-child .cell {

padding-left: 20px;

}

**偶数行颜色**

.el-table\_\_body {

tr.el-table\_\_row--striped {

td {

background:rgba(249,249,249,1); // 偶数行条纹颜色

}

// 偶数行点击当前行时属性

&.current-row td {

background-color: rgba(226,243,255,1);

}

}

// hover时颜色

tr:hover > td {

background-color: rgba(242,248,255,1);

}

// el-table配置highlight-current-row属性 点击行时有current-row属性

tr.current-row > td {

background-color: rgba(226,243,255,1);

}

}

**组件数据说明**

tableData: Array, // 表格数据

columnsType: String, // 区分columns

state: Function, // 操作事件

hasPagination: { type: Boolean, default: true }, // 控制是否显示分页

maxHeight: { type: String, default: "710" },

oprationWidth: { type: Number, default: 180 },

\_columns: { // 表头数据及prop

type: Array,

default() {

if (this.columnsType == "salesOrder") {

return relation.salesOrder;

} else if (this.columnsType == "mainProductionPlan") {

return relation.mainProductionPlan;

} else if (this.columnsType == "selfMade") {

return relation.selfMade

} else if (this.columnsType == "processTask") {

return relation.processTask

} else if (this.columnsType == "equipmentWorkOrder") {

return relation.equipment

}

}

},

fixed: Array, // 固定列说明 fixed[2, 10] 固定左边2列和第10列

pageObj: Object, // 页码改变 父组件刷新

updateCallback: Function, // 回调父组件刷新数据

## $route.name

可在使用时取值 在注册路由时定义 可跟path保持一致

path: '/projectList',

component: Layout,

hidden: true,

**name**: 'mes-base-projectList',

## form表单resetFields()

对整个表单进行重置，将所有字段值重置为初始值，并移除校验结果。

this.$refs.form.resetFields()

## watch使用方法

watch: {

'form.value': {

handler(val) {

console.log('form.value', val)

}

},

'types': {

deep: true,

handler(val) {

console.log('types', val[0].key)

}

},

},

## v-for使用key

为了给Vue一个提示，以便它能跟踪每个节点的身份，从而复用和重新排序现有元素，需要给每项提供一个唯一的key属性。建议尽可能的在使用v-for时提供key。

<div v-for=”item of items” :key=”item.id”>

## refs嵌套使用

this.$refs.editbomValidateForm.$refs.bomForm.validate((valid) => {})

editbomValidateForm是组件的ref，bomForm是组件内form表单的ref。

## el-tree slot-scope

el-tree使用slot-scope 不能用template 需使用span 不像el-table

## 文本溢出css样式

.custom-tree-node {

overflow: hidden;

text-overflow: ellipsis; //文本溢出显示省略号

white-space: nowrap;

}

## tree组件设置默认高亮

node-key="**locationCode**" –设置node-key

this.$refs.tree.setCurrentKey(res.data[0].**locationCode**);

## v-if v-show

v-if

指令必须将它添加到一个元素上，如果想切换多个元素，可以把<template>元素当做一个不可见的包裹元素，最终的渲染结果不包含<template>。

使用key管理可复用的元素，如果允许用户在不同登录方式之间切换，且清除用户已经输入的数据，因为使用的相同的元素，下面代码在切换时将不会清除数据。

<template v-if="loginType === 'username'">  
 <label>Username</label>  
 <input placeholder="Enter your username">  
</template>  
<template v-else>  
 <label>Email</label>  
 <input placeholder="Enter your email address">  
</template>

Vue提供了一种方式，添加唯一值key属性。

<template v-if="loginType === 'username'">  
 <label>Username</label>  
 <input placeholder="Enter your username" key="username-input">  
</template>  
<template v-else>  
 <label>Email</label>  
 <input placeholder="Enter your email address" key="email-input">  
</template>

v-if是真正的条件渲染，它会确保在切换中条件块内的事件监听器和子组件适当的被销毁和重建。

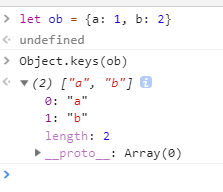
v-show

不同的带有v-show的元素始终会被渲染并保留在DOM中，v-show只是简单的切换元素css属性display。v-show不支持<template>元素和v-else。

一般来说，v-if有更高的切换开销，而v-show有更高的初始渲染开销。如果需要非常频繁的切换，使用v-show更好。如果在运行中条件很少改变，使用v-if较好。

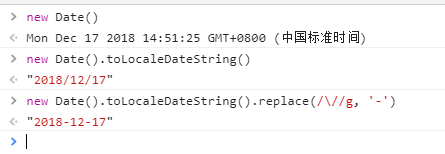
## 判断对象是否为空

ES6方法Object.keys()，返回值是对象中属性名组成的数组



## toLocaleDateString

根据本地时间把Date对象的日期部门转换为字符串，并返回结果。



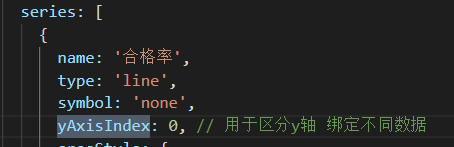
## 根据分辨率设置字体大小方法

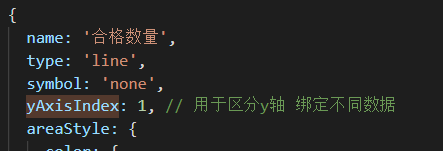
let scale = Math.max(document.body.offsetWidth / 1200, 1) 倍数

font-size: 14px \* scale

## echarts 多个y轴区分数据显示

yAxisIndex 使用的y轴的index，在单个图表实例中存在多个y轴的时候有用





## 表格组件格式化内容 formatter

label: '检测结果',

prop: 'qcResult',

formatter: (row, column, **cellValue**, index) => {

if (**cellValue** >= 0) {

return cellValue + '' === '1' ? '合格' : '不合格'

} else {

return ''

}

}

无需在组件中做耦合处理，传入组件前做处理，针对单个内容。