# dm\_list\_data配置文档

1. 远程引用

<script src="http://qn-static.dmagic.cn/util.1.1.0.js"></script>

  <script src="http://qn-static.dmagic.cn/vue-dm-components.2.0.1.js"></script>

1. 基础配置

|  |  |  |  |
| --- | --- | --- | --- |
| **属性名** | **属性类型** | **属性值说明** | **参考配置** |
| cf | 配置列表对象 | 用于配置列表，详细配置见下面cfList配置 | :cf=”cfList” cfList:{} |
| after-add | 事件 | 新增成功之后的回调事件，返回新增前的数据，以及新增的数据 | @after-add=”afterAdd(newdata,olddata)” |
| after-modify | 事件 | 修改成功之后的回调事件，返回修改前的数据，以及修改后的数据 | @after-modify=”afterModify(newdata,olddata)” |
| after-delete | 事件 | 删除成功之后的回调事件，返回要删除的数据 | @after-delete=”afterDelete(data)” |
| after-search | 事件 | 查询成功之后的回调事件，返回列表数据 | @after-search=”afterSearch(data)” |
| after-show-Dialog-Modify | 事件 | 打开修改弹窗之前触发的事件，返回当前列数据 | @after-show-Dialog-Modify  =”showModify” |
| after-show-Dialog-Add | 事件 | 打开新增弹窗之前触发的事件 | @after-show-Dialog-Add  =”showAdd” |
| after-show-Dialog-Detail | 事件 | 打开详情弹窗之前触发的事件 | @after-show-Dialog-Detail  =”showDetail” |
| bacth-btn-click | 事件 | 自定义批量操作按钮触发事件,返回eventType 事件类型，如果有选中数据，返回selection选中数据 | @bacth-btn-click=”customEven(eventType, selection)” |
| single-btn-click | 事件 | 自定义单项操作按钮触发事件，返回eventType 事件类型以及row列数据 | @single-btn-click=”customEven(eventType,row)” |
| getDataList | 方法 | 调用该方法刷新列表数据，通过给列表组件添加ref可以调用到此方法 |  |

3.cfList配置

|  |  |  |
| --- | --- | --- |
| **属性名** | **属性值说明** | **参考配置** |
| listIndex | 当前页面路由名，保存在vuex，用于聚焦 | listIndex: "list\_demo" |
| focusMenu | 是否进行路由聚焦 | focusMenu: true |
| pageSize | 列表当前页显示条数 | pageSize:10 |
| expand | 列表是否需要展开行，设置为true的话，需要配合expands属性一起使用 | expand: true,  expands: [{              label: "分类名称",              prop: "articleCategory",               width: 150,              formatter: function(rowData) {              },  slot: "slot\_form\_expand\_articleTitle"            }]  <dm\_list\_data :cf="cfList" >  <template v-slot:slot\_form\_expand\_articleTitle="{row}">          {{row}}        </template>  </dm\_list\_data> |
| expands | 列表展开行之后显示的数据，支持slot和formatter |
| url | 列表获取数据所调用接口 | url: {            list: "/crossList?page=info\_piece", //列表接口            add: "/crossAdd?page=info\_piece", //新增接口            modify: "/crossModify?page=info\_piece", //修改接口            detail: "/crossDetail?page=info\_piece",            delete: "/crossDelete?page=info\_piece" //删除接口          }, |
| breadcrumb | 列表面包屑的配置，是一个对象数组，对象的value属性代表显示内容，可为对象添加path设置面包屑跳转路由 | breadcrumb: [            { value: "首页", path: "/listHome" },            { value: "其他数据" },            { value: "文章管理" }          ], |
| cfElTable | 可以通过配置该对象，为table组件设置新的配置 | cfElTable: {            "default-sort": { prop: "CreateTime", order: "descending" }          }, |
| dynamicDict | 动态字典配置，是一个对象数组，其中每个对象有page(需连接的数据表),populateColumn(获取到的数据),idColumn(以该属性去查找),idColumn2(数据表中的属性key) | dynamicDict: [            {              page: "tangball\_article\_category",              populateColumn: "categoryDoc",              idColumn: "articleCategory",              idColumn2: "P1"            }          ], |

|  |  |  |
| --- | --- | --- |
| **属性名** | **属性值说明** | **参考配置** |
| singleBtns | 设置单项操作按钮的配置，如果有自定义事件，需要配合列表的@single-btn-click方法进行操作，默认会显示详情，修改，删除三个按钮，如果需要添加其他按钮，则需要重新配置详情，修改，删除按钮 | singleBtns: {  addon: [              util.cfList.sBtns.detail,              util.cfList.sBtns.modify,              util.cfList.sBtns.delete,  {//自定义操作按钮配置                title: "单项操作（圆形按钮）",                eventType: "singleOP1",                cfElBtn: {                  circle: true,                  icon: "el-icon-user-solid"                }              },  {//自定义跳转页面配置                uiType: "link",                text: "新窗口打开页面",                target: "\_blank",                urlFormatter: function(row) {                  return `http://www.baidu.com?id=${row.name}`;                },                url: "http://www.baidu.com?id="              }  ]  } |
| batchBtns | 设置单项操作按钮的配置，如果有自定义事件，需要配合列表的@bacth-btn-click方法进行操作，默认会显示新增,删除选中两个按钮，如果需要添加其他按钮，则需要重新配置新增,删除选中按钮 | batchBtns: {  addon: [              util.cfList.bBtns.add,  util.cfList.bBtns.delete,      {//自定义批量操作按钮，需要选中数据                text: "批量操作（需选数据）",                eventType: "bacthOP1",                needSelect: true,                cfElBtn: {                  icon: "el-icon-user-solid"                }              },  //自定义批量操作按钮，不需要选中数据              { text: "其他操作（不需选中数据）", eventType: "bacthOP2" },  //自定义页面跳转按钮              {                uiType: "link",                text: "新窗口打开页面",                target: "\_blank",                url: "http://www.baidu.com",                cfElBtn: {                  type: "primary",                  icon: "el-icon-user-solid"                }              },  //设置插槽              { uiType: "slot", slot: "slot\_in\_toolbar" }  ]  tips: { //设置提示语              text: "提示语123"              style:{"color":"#3a0"}            }  } |
| isShowSearchForm | 是否显示查询列表，默认true | isShowSearchForm:true |
| isShowToolBar | 是否显示工具栏（批量操作按钮）,默认true | isShowToolBar:true |
| isShowCheckedBox | 列表是否显示多选框,默认true | isShowCheckedBox:true |
| isShowOperateColumn | 列表是否显示操作栏,默认true | isShowOperateColumn:true |
| isShowPageLink | 是否显示分页，默认为true | isShowPageLink:true |
| customDetail | 是否启用自定义详情弹窗，默认false，如果设置为true的话，需要用插槽配置自定义详情弹窗显示内容 | customDetail: true  <template v-slot:customDetail="{detailData}">          <!-- 自定义案件详情弹窗 -->          <case\_detail\_dialogs :caseMsg="detailData"></case\_detail\_dialogs>        </template> |
| findJsonDefault | 列表查询接口数据默认参数（findJson） | findJsonDefault:{        $or:[{status:8},{status:15}]      },  //过滤案件状态为8或者案件状态为15的数据 |
| formDataAddInit | 列表进行新增时默认携带数据 | formDataAddInit: {        //新增案件时默认添加创建人属性        createPerson: Number(localStorage.userId)      }, |
| cfSearchForm | 查询列表的配置，可以通过该对象的watch属性设置传入监听器，当查询数据变化时响应变化 | cfSearchForm: {            watch: {               articleCategory(newVal, oldVal) {                console.log("complete1111变动");               }            }          }, |

|  |  |  |
| --- | --- | --- |
| **属性名** | **属性值说明** | **参考配置** |
| cfDialogForm | 新增修改弹窗的提示语配置，可以通过配置该对象的tips属性设置新增修改弹窗的提示语 | cfDialogForm: {            tips: {              text: "1,表单的提示语1<br/> 2,表单的提示语2",              style: { "padding-left": "200px" }            }          }, |
| cfForm | 新增修改弹窗的配置，通过该对象的watch属性可以给新增修改弹窗传入监听器，col\_span属性可以设定弹窗中显示列所占宽度，默认24，一半则配置12 | cfForm: {            col\_span: 12            watch: {                articleCategory(newVal, oldVal) {                  }            },          }, |

4.cfList的columns属性配置（columns是列表的每一列数据配置,是一个对象数组,下面是该数组中对象的属性配置,参考配置,columns:[{},{}]）

|  |  |  |
| --- | --- | --- |
| **属性名** | **属性值说明** | **参考配置** |
| **label** | 当前列的表头名 | label: "案件状态" |
| prop | 当前列在返回数据中的属性名 | prop: "status" |
| sortable | 当前列是否需要支持排序功能，将其设置为”custom”代表开启排序功能 | sortable: "custom", |
| width | 当前列所占宽度 | width: 145 |
| filters | 配置当前列是否开启筛选，需要和columnKey配合使用，是一个对象数组，对象中的text属性表示显示的文字，value表示其代表的数据 | filters: [                { text: "待立案", value: 1 },                { text: "已结案", value: 2 }              ],              columnKey: "status" |
| columnKey | 设定当前列筛选数据根据的属性名 |
| requireProp | 当前列的依赖字段，默认列表数据只会返回有设置的数据，如果需要额外依赖字段，则通过该字段配置 | requireProp: ["collaborator",'createPerson'], |
| formatter | 当前列数据的显示格式，该属性值是一个方法，接受当前行的数据 | formatter: function(row) {            if (row.trialDate) {              return moment(row.trialDate).format("YYYY-MM-DD");            } else {              return "";            }          } |
| Slot | 当前列是否需要通过插槽显示，设定该属性为插槽名则显示插槽 | slot: "slot\_column\_CreateTime"  <dm\_list\_data :cf="cfList" >  <template v-slot\_column\_CreateTime="{row}">         {{row}}    </template>  </dm\_list\_data> |

5.cfList的detailItems属性配置（detailItems是列表的详情弹窗数据配置,是一个对象数组,下面是该数组中对象的属性配置,参考配置,detailItems:[{},{}]）

|  |  |  |
| --- | --- | --- |
| **属性名** | **属性值说明** | **参考配置** |
| **label** | 当前行名 | label: "案件状态" |
| prop | 当前行对应的属性名 | prop: "status" |
| formatter | 当前行数据的显示格式，该属性值是一个方法，接受当前行的数据 | formatter: function(row) {            if (row.trialDate) {              return moment(row.trialDate).format("YYYY-MM-DD");            } else {              return "";            }          } |
| Slot | 当前行是否需要通过插槽显示，设定该属性为插槽名则显示插槽 | slot: "slot\_detail\_item\_matchInfo"  <dm\_list\_data :cf="cfList" >  <template v-slot:slot\_detail\_item\_matchInfo="{row}">         {{row}}    </template>  </dm\_list\_data> |

这里要有一个要注意的地方是列表配置可以通过数据字典的方式让一些显示会员id的转化成显示会员姓名，而详情这里无法使用

有一个常用的dm\_ajax\_populate组件可以解决这个问题，配置方法如下：

 <template v-slot:slot\_detail\_item\_matchId="{row}">

        <dm\_ajax\_populate :id="row.matchId" populateKey="matchName" page="tangball\_match">

          <template v-slot:default="{doc}">

            <div class v-if="doc && doc.P1">

              {{doc.P1}}

              (

              {{doc.matchName}})

            </div>

          </template>

        </dm\_ajax\_populate>

      </template>

该组件的id属性是所根据查找的数据，page是要根据查找的接口名

6.cfList的formItems，searchFormItems属性配置（formItems是列表的修改弹窗数据配置,searchFormItems是列表的查询数据配置,都是是一个对象数组,下面是该数组中对象的属性配置,参考配置,formItems:[{},{}],searchFormItems:[{},{}]）

|  |  |  |
| --- | --- | --- |
| **属性名** | **属性值说明** | **参考配置** |
| **label** | 当前行名 | label: "案件状态" |
| prop | 当前行对应的属性名 | prop: "status" |
| cfForm | 设置子表单数据配置，col\_span属性设置子表单每一列所占宽度，默认24，占一整行，formItems  属性配置子表单显示的数据，配置方式与cfList.formItems一致 | {          label: "基本信息",          prop: "",          default: {},          cfForm: {            col\_span: 12,            formItems: [        {          label: "案号",          prop: "caseId"        },  ]} |
| ajax | 根据接口数据渲染的下拉框，需要配合type为select属性一起使用 | type: "select",          ajax: {            url: "/crossList?page=lawyer\_area",            keyLabel: "name",            keyValue: "P1"          } |
| type | 设定当前输入框的类型，如select，radio，  默认是普通text输入框 | type:”select” //下拉框  options: [      { label: "已支付", value: 2 },      { label: "未支付", value: 1 }    ]  type:”select\_list\_data” //弹窗选择列表  type:”radio” //单选框  type:”checkbox” //多选框  type:”textarea” //文本域  type:”date” //日期选择  type:”dateTime” //日期时间选择  type:”time\_period” //时间段选择  type:”slider” //百分比滑块  type:”vueJsonEditor” //vue-json编辑器  type:”jsonEditor” //普通json编辑器  type:”collection” //collection  type:”upload” //图片上传  type:”editorTM” //tiny\_mce富文本编辑器  type:”editor” //普通富文本编辑器  type:”input\_find\_vague” //模糊查询文本框  type:”password” //密码输入框  type:”number” //数字输入框  type:”numberRange” //numberRange数字范围  type:”ajax\_populate” /ajax\_populate  type:”text” //普通文本输入框 |
| rules | 定义的输入框的规则 | [{ required: true, message: "案件名称不能为空" }] |
| term | 设置依赖于指定条件显示 | term: { matchForm: 2 }, //只有在赛事类型为2的时候才会显示 |
| col\_span | 设置输入框所占的大小，默认24一整行  占据一半则设置12 | col\_span:12 |
| tips | 设置提示语句 | tips: "图片尺寸比例最好保持在1:0.8，建议尺寸：宽500px，高400px" |
| slot | 当前输入框是否需要通过插槽显示，设定该属性为插槽名则显示插槽 | slot: "slot\_form\_item\_memberIdList"  <dm\_list\_data :cf="cfList" >  <template v-slot:slot\_form\_item\_memberIdList="{formData}">         {{formData}}    </template>  </dm\_list\_data> |