

TUI使用文档 V1.1.9

使用准备

基于jQuery, bootstrap, react, simditor(可选)

- [jQuery](http://www.qintrend.net/resources/backend-theme/js/jquery-1.10.1.min.js): <http://www.qintrend.net/resources/backend-theme/js/jquery-1.10.1.min.js>

- [bootstrap](#)

css: <http://www.qintrend.net/resources/bs-theme/bootstrap/css/bootstrap.min.css>

js: <http://www.qintrend.net/resources/bs-theme/bootstrap/js/bootstrap.min.js>

- [react](#)

react: <http://www.qintrend.net/resources/jsLib/react/0.14/react.min.js>

react-dom: <http://www.qintrend.net/resources/jsLib/react/0.14/react-dom.min.js>

- [simditor](#): <http://www.qintrend.net/resources/jsLib/trend/simditor/simditor.all.min.js>

- [tui](#)

css: <http://www.qintrend.net/resources/jsLib/trend/tui/1.1/css/tui.all.min.css>

js: <http://www.qintrend.net/resources/jsLib/trend/tui/1.1/build/tui.all.min.js>

表格 – TUI.table 提供两个参数`options`和`entity`(可选), 说明如下

`TUI.table(options, entity);`

options 表格选项

属性	说明
<code>container</code> string	容器ID
<code>formId</code> string	表单ID, 可选, 不填会生成默认ID

method string	远程请求方式 GET/POST，默认GET方式请求
url string	远程调用方法 返回值：对象，必须包含rows和rowCount属性，分别表示数据集合和总记录数
columns array	定义列属性 返回值：数组，每列为一个对象 事例： <pre> columns: [{ // 列名称 text: 'ID', // 列对应后台的字段名 支持多级调用 field: 'id', // 该方法可对该列值进行特殊操作 // 接受一个参数，为当前对象 handle: function(obj) { }, // 是否开启checkbox选项，默认false不开启 // 此选项一般用在主键列上，并配合toolbar使用 checkbox: true }] </pre>

rowHandles object

行操作，目前支持删除和查看

返回值：对象

事例：

```
rowHandles: {
  // 删除，接收一个参数，为当前记录对象
  delete: function(obj) {
    return {
      method: 'get',
      url: url,
      // 回调函数，如果删除成功请返回success
      callback: function(result) {
        return 'success';
      }
    };
  },
  // 查看，接收一个参数，为当前记录对象
  find: function(obj) {
    return {
      // 默认get方式
      method: 'get',
      url: '/teacher/get/' + obj.objectId,
      // 可选，回调函数
      // 如果你需要对得到的对象进行操作可以使用callback
      callback: function(result) {
        return result;
      }
    };
  },
  // 自定义行操作
  custom: [
    {
      // 操作名称
      text: '自定义操作',
      // 图标class 可选
      iconClass: 'fa fa-cog',
      // 处理方法，接受当前row对象
      handle: function(row) {
        alert('当前id是' + row.objectId);
      }
    }
  ]
}
```

searchbar array

筛选条件工具栏

返回值：数组

事例：

searchbar: [

```
    {
      // 类型，默认text
      // 目前支持的类型: input select radio
      type: 'select',
      // 筛选条件名称
      text: '项目类型',
      // 对应的后台字段
      field: 'companyOrPersonal',
      // 当type为select或radio时，需要用options填充值
      // select 支持远程加载
      options: [
        { text: '不限' },
        { text: '赣核盈', value: 'company' },
        { text: '赣核贷', value: 'personal' }
      ]
    }
  ]
```

当type为select时，options有三种赋值方法

- 1.本地加载，也就是事例中所写的
- 2.远程加载，options值为远程url，返回集合，集合内的对象要有text / value属性
- 3.远程加载2，options值为对象，有三个属性：url，textField，valueField，
这种方法更灵活，可以自定义text / value对应的属性名

toolbar array	<p>功能性工具栏 目前支持button和checkbox</p> <p>返回值：数组</p> <p>事例：</p> <pre> toolbar: [// button用法 { // 按钮名称 text: '删除', // class className: 'btn btn-danger', // 处理方法 handle: function() { }, }, // checkbox用法 { text: '点我', type: 'checkbox', id: checkbox id, name: checkbox name, // 是否选中 默认false checked: true, },] </pre>
pagination boolean	是否分页，默认true分页
maxSize int	每页最大显示记录数，默认10条

entity 实体类选项

属性	说明
key string	主键字段名

属性	说明
fields array	<p>实体字段，添加或查看记录时显示的字段</p> <p>目前支持的类型：input(文本) select(下拉框) radio(单选按钮) textarea(大文本) editor(文本编辑器)</p> <p>事例：</p> <pre> fields: [{ // 字段名称 text: '性别', // 对应的后台字段名 field: 'gender', // 类型 默认text // 目前支持的类型: input select radio type: 'select', // type为select或radio时需要使用options填充值 options: [{ text: '男', value: 'MALE' }, { text: '女', value: 'FEMALE' }, { text: '未知', value: 'UNKNOWN' }], // 验证器 validators: { // 非空验证 message属性可不写 notEmpty: { message: 'XX不能为空' }, // 远程验证，返回json格式 // {valid: true} true通过 false不通过 remote: { url: '', data: { key: 'gender', value: 'gender' }, message: '性别填写不正确' } }, readOnly: false, disabled: false }] </pre> <p>当type为select时，options有三种赋值方法</p> <p>1 本地加载 也就是事例中所写的</p>

属性	说明
create object	<p>创建实体</p> <p>事例：</p> <pre> create: { method: 'POST', url: '/teacher/create', // 接收一个参数，返回结果，如果创建成功请返回success callback: function(result) { return 'success'; } }</pre>
update object	<p>更新实体</p> <p>事例：</p> <pre> update: { method: 'POST', url: '/teacher/update', // 接收一个参数，返回结果，如果创建成功请返回success callback: function(result) { return 'success'; }, // 可选，可以自定义是否显示修改按钮，返回success显示 condition: function(entityData) { return 'success'; } }</pre>
custom array	<p>自定义实体操作</p> <p>事例：</p> <pre> custom: [{ // 操作名称 text: '自定义操作', // 图标class 可选 className: 'fa fa-cog', // 处理方法，接受当前row对象 handle: function(row) { alert('当前id是' + row.objectId); // 返回success则会返回并刷新列表 return 'success'; } }]</pre>

模态框 — TUI.Modal

```
var modalProps = {  
    id: 'id', // 模态框ID, 可选  
    title: '标题',  
    content: '内容', // 模态框内容, 可以输入html  
    // 确定事件  
    confirm: function() {  
        // 返回success关闭模态框  
        return 'success';  
    }  
};  
  
TUI.Modal(modalProps);
```

弹出提示信息框

```
TUI.success('成功');  
TUI.danger('失败');  
TUI.warning('警告');  
TUI.info('信息');
```

加载提示框

```
显示  
TUI.Loading({ text: '正在加载中' });  
关闭  
TUI.Loading({ className: 'hide' });
```

工具类

```
格式化日期 — 传入毫秒  
TUI.Utls.dateFormat(date, 'yyyy-mm-dd hh:mm:ss');  
TUI.Utls.dateFormat(date, 'yyyy-mm-dd');
```