

# Mint UI Unit02

## 1. 表单组件

### • Field 组件

Field 组件用于实现表单控件，其语法结构是：

```
<mt-field type="输入框的类型" label="标签"
  placeholder="占位内容"
  state="检测状态"
  v-model="变量名称"
  :attr="原生属性"
  readonly
  disabled
  disableClear>
</mt-field>
```

输入框的类型有：

- text，单行文本框
- password，密码框
- textarea，多行文本框
- number，数字
- tel，电话
- url，URL地址
- date，日期
- datetime-local，日期和时间

state 属性表示输入框的状态，其值有：

- error，表示错误
- success，表示成功
- warning，表示警告

:attr 属性用于控制表单控件的原生属性，对象类型，如：

```
<mt-field type="password"
  label="密码"
  :attr="{autocomplete:'off',maxLength:'10'}">
</mt-field>
```

disabled 属性用于控制表单控件是否为禁用状态，布尔类型，默认为 false

readonly 属性用于控制表单控件是否为只读状态，布尔类型，默认为 false

disableClear 属性用于控制表单控件是否禁用 clear 按钮，布尔类型，默认为 false

如果为表单控件添加失去焦点、获取焦点的事件，其语法结构是：

```
<mt-field @blur.native.capture="事件名称"></mt-field>
```

```
<mt-field @focus.native.capture="事件名称"></mt-field>
```

Vue.js的事件修饰符<https://cn.vuejs.org/v2/guide/events.html>

## • **checklist** 组件

`checklist` 组件用于实现复选框列表，其语法结构是：

```
<mt-checklist title="标题" align="对齐方式(left|right)"
               :option="选项列表"
               v-model="变量名称">
</mt-checklist>
```

`:options` 属性用于控制复选框的列表项，数据类型为数组，数组中既可以是字符串，也可以是对象，如：

```
// 数组形态
['文本', '文本', ...]

// 对象形态
[
  {
    label: '标签文本',
    value: '值',
    disabled: 是否禁用(true|false)
  }
]
```

`v-model` 指令绑定的变量的数据类型必须为数组类型（因为其有可能要选择多个，所以为数组类型）

必须为 `mt-checklist` 组件赋予 `v-model`、`:options` 属性

```
Go Run Terminal Help Checklist.vue - demo - Visual Studio Code
▼ Register.vue ▼ App.vue ▼ Checklist.vue × ▼ Radio.vue JS index.js
src > components > MintUI > ▼ Checklist.vue > {} "Checklist.vue" > template > div > mt-checklist

3       <p>{{loves}}</p>
4       <mt-checklist title="爱好"
5       :options="options" v-model="loves"
6       align="left">
7     </mt-checklist>
8   </div>
9 </template>
10 <script>
11 export default {
12   data(){
13     return {
14       loves:[],
15       options:[
16         {
17           label:'看电视',
18           value:'1'
19         },
20         {
21           label:'看电影',
22           value:'2'
23         },
24         {
25           label:'看书',
26           value:'3'
27         },
28         {
29           label:'睡觉',
30           value:'4'
31         }
32       ]
33     }
34   }
35 }
36 </script>
```

## • Radio组件

Radio 组件用于实现单选框列表，其语法结构是：

```
<mt-radio title="标题" align="对齐方式(left|right)"
  :options="选项列表" v-model="变量名称">
</mt-radio>
```

`v-model` 指令绑定的变量的数据类型为字符类型

`:options` 属性用于控制单选框的列表项，数据类型为数组，数组中既可以是字符串，也可以是对象，如：

```
//数组形态
['文本', '文本', ...]

//对象形态
[
  {
    label: '标签文本',
    value: '值',
    disabled: 是否禁用(true|false)
  }
]
```

```
Go Run Terminal Help Radio.vue - demo - Visual Studio Code
▼ Register.vue ▼ App.vue ▼ Checklist.vue ▼ Radio.vue x JS index.js
src > components > MintUI > ▼ Radio.vue > {} "Radio.vue" > template > div > mt-radio
1  <template>
2    <div>
3      <p>{{ sex }}</p>
4      <mt-radio title="性别" :options="options" v-model="sex" align="left">
5    </mt-radio>
6  </div>
7 </template>
8 <script>
9 export default {
10   data(){
11     return {
12       sex:'',
13       options:[
14         {
15           label:'男',
16           value:'0'
17         },
18         {
19           label:'女',
20           value:'1'
21         },
22         {
23           label:'保密',
24           value:'2'
25         }
26       ]
27     }
28   }
29 }
30 </script>
```

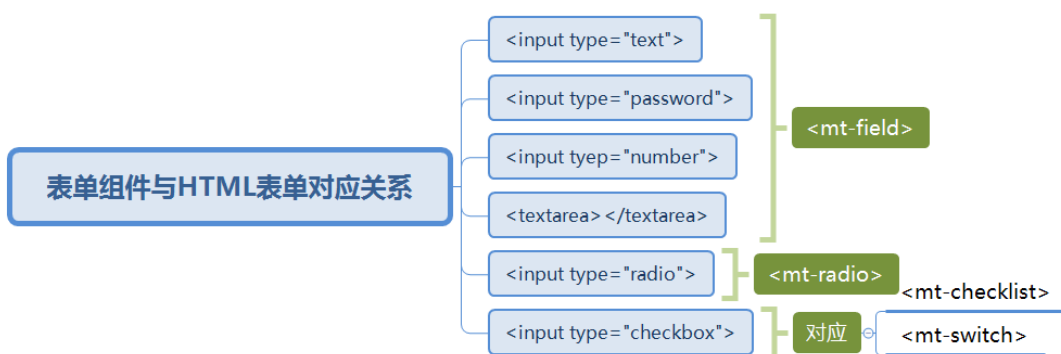
## • Switch 组件

`switch` 组件用于实现开关，其语法结构是：

```
<mt-switch v-model="变量名称" disabled>
  ...
</mt-switch>
```

v-model 指令绑定的变量的数据类型为布尔类型

```
Go Run Terminal Help Switch.vue - demo - Visual Studio Code
Register.vue App.vue Checklist.vue Radio.vue Switch.vue X
src > components > MintUI > Switch.vue > {} "Switch.vue" > script
1 <template>
2   <div>
3     <p>{{allow}}</p>
4     <mt-switch v-model="allow">
5       是否接收消息提示
6     </mt-switch>
7   </div>
8 </template>
9 <script>
10 export default {
11   data(){
12     return {
13       allow:false
14     }
15   }
16 }
17 </script>
```



## 2.CSS 组件

### • Cell 组件

Cell 组件用于实现单元格，其语法结构是：

```
<mt-cell title="标题" label="备注信息" value="内容" icon="图标"
  isLink
  to="URL地址">
  ...
</mt-cell>
```

icon 属性用于控制单元格图标，其值可为 more、back 或字体图标类名称

单元格内的图像可以通过 slot="icon" 来修改单元格图标

isLink 属性用于控制单元格是否为可点击的链接，布尔类型，默认为 true

to 属性用于控制链接单元格的目标路由，字符类型

```
Go Run Terminal Help Cell.vue - demo - Visual Studio Code
▼ About.vue ▼ Profile.vue ▼ Cellswipe.vue ▼ MessageBox.vue ▼ NavBar.vue ▼ Home.vue ▼ Cell.vue × JS index.js
src > components > MintUI > ▼ Cell.vue > {} "Cell.vue" > template > div > mt-cell
1 <template>
2   <div>
3     <mt-cell title="蓝牙" label="备注信息">
4       <mt-switch v-model="blue"></mt-switch>
5     </mt-cell>
6     <mt-cell title="蓝牙" label="备注信息" icon="more">
7       <mt-switch v-model="blue"></mt-switch>
8     </mt-cell>
9     <mt-cell title="蓝牙" label="备注信息" icon="back">
10      <mt-switch v-model="blue"></mt-switch>
11    </mt-cell>
12    <mt-cell title="蓝牙">
13      <mt-switch v-model="blue"></mt-switch>
14      
15    </mt-cell>
16    <mt-cell title="红包" isLink to="/">
17      
18    </mt-cell>
19  </div>
20 </template>
21 <script>
22 export default {
23   data(){
24     return {
25       blue:true
26     }
27   }
28 }
29 </script>
```

## • cell swipe 组件

cell swipe 组件用于实现可滑动的单元格，其语法结构是：

```
<mt-cell-swipe title="标题" label="备注信息" value="内容" icon="图标"
  isLink to="URL地址"
  :left="左侧按钮组"
  :right="右侧按钮组">
  ...
</mt-cell-swipe>
```

:left 和 right 属性分别用于控制滑动单元格的按钮组，数据类型为数组，结构如下：

```
:right = "[
  {
    content: '按钮显示文本',
    style: {属性名称:值,属性:值,...},
    handle:()=>{ ... }
  },
  ...
]"
```

针对 [Intervention] Ignored attempt to cancel a touchmove event with cancelable=false, for example because scrolling is in progress and cannot be interrupted. 错误提示的解决方案：

在对应的组件文件内添加以下样式：

```
<style>
  html,body{
    touch-action:none;
  }
</style>
```

```
Go Run Terminal Help Cellswipe.vue - demo - Visual Studio Code
▼ About.vue ▼ Profile.vue ▼ Cellswipe.vue × ▼ MessageBox.vue ▼ Navbar.vue ▼ Home.vue ▼ Cell.

src > components > MintUI > ▼ Cellswipe.vue > ☐ template > ☐ div > ☐ mt-cell-swipe

1  <template>
2      <div>
3          <mt-cell-swipe title="10086" :right="[
4              {
5                  content:'隐藏提醒',
6                  style:{background:'#00f',color:'#fff'},
7                  handler:()=>{
8                      this.$toast('aaa');
9                  }
10             },
11             {
12                 content:'删除',
13                 style:{background:'#f00',color:'#fff'},
14                 handler:()=>{
15                     this.$toast('bbbb');
16                 }
17             }
18         ]">
19             请您为10分满意进行评分:
20         </mt-cell-swipe>
21     </div>
22 </template>
23 <style>
24     html,body{
25         touch-action:none;
26     }
27 </style>
28
```

## • Navbar 组件

Navbar 组件用于实现顶部选项卡，其语法结构是：

```
<mt-navbar fixed v-model="变量名称">
  <mt-tab-item id="当前选项卡ID">
    ...
  </mt-tab-item>
  <mt-tab-item id="当前选项卡ID">
    ...
  </mt-tab-item>
  ...
</mt-navbar>
```



```
Go Run Terminal Help Navbar.vue - demo - Visual Studio Code
▼ About.vue ▼ Profile.vue ▼ Cellswipe.vue ▼ MessageBox.vue ▼ Navbar.vue X ▼ Home.vue ▼ Cell.vue
src > components > MintUI > ▼ Navbar.vue > {} "Navbar.vue" > template > div > mt-navbar
1 <template>
2   <div>
3     <p>{{ selected }}</p>
4     <mt-navbar v-model="selected">
5       <mt-tab-item id="a">推荐</mt-tab-item>
6       <mt-tab-item id="b">生活</mt-tab-item>
7       <mt-tab-item id="c">娱乐</mt-tab-item>
8       <mt-tab-item id="d">汽车</mt-tab-item>
9     </mt-navbar>
10  </div>
11 </template>
12 <script>
13 export default {
14   data(){
15     return {
16       selected: 'a'
17     }
18   }
19 }
20 </script>
```

## 3. JS 组件

### • MessageBox 组件

MessageBox 组件用于实现弹出式提示框，其语法结构是：

```
// 简捷方式
this.$messagebox("标题", "提示信息")
```

MessageBox 还提供了 `alert()`、`confirm()` 及 `prompt()` 三个方法，它们都返回 `Promise`，语法结构分别是：

```
// alert() 方法
this.$messagebox.alert("提示信息", "标题", 选项)

// confirm() 方法
this.$messagebox.confirm("提示信息", "标题", 选项)

// prompt() 方法
this.$messagebox.prompt("提示信息", "标题", 选项)
```

选项参数用于控制提示框的相关配置，对象类型，包括：

- `showConfirmButton`，控制是否显示确认按钮，默认 `true`
- `confirmButtonText`，控制确认按钮的文本，默认为 确认
- `confirmButtonClass`，控制确认按钮的 CSS 类名称
- `confirmButtonHighLight`，控制确认按钮文本是否高亮显示，默认为 `false`，需重新定义 CSS 类，如：

```
<style>
  .mint-msgbox-confirm:active {
    color: #0563a9;
    font-weight: bold;
  }

  .mint-msgbox-btn:active{
    background-color:#fff;
  }
</style>
```

- `showCancelButton`，控制是否显示取消按钮，默认 `true`
- `cancelButtonText`，控制取消按钮的文本，默认为 取消
- `cancelButtonClass`，控制取消按钮的 CSS 类名称
- `cancelButtonHighLight`，控制取消按钮文本是否高亮显示，默认为 `false`，需重新定义 CSS 类，如：

```
<style>
  .mint-msgbox-cancel:active {
    color: #0563a9;
    font-weight: bold;
  }

  .mint-msgbox-btn:active{
    background-color:#fff;
  }
</style>
```

- `closeOnClickModal`，控制是否在单击遮罩层时关闭提示框，默认为 `true`
- `closeOnPressEscape`，控制是否通过键盘 ESC 按钮关闭提示框，默认为 `true`
- `lockScroll`，属性是否锁定滚动条，默认为 `false`
- `inputType` 属性用于控制输入框的类型，默认为 `text`
- `inputPlaceholder` 属性用于控制输入框的占位符

```
Go Run Terminal Help Messagebox.vue - demo - Visual Studio Code
▼ About.vue ▼ Profile.vue ▼ Cellswipe.vue ▼ Messagebox.vue X ▼ Navbar.vue ▼ Home.vue ▼ Cell.vue JS index.js
src > components > MintUI > ▼ Messagebox.vue > {} "Messagebox.vue" > script > methods > prompt > then() callback
1 <template>
2   <div>
3     <p>1111</p>
4     <p>1111</p>
5     <p>1111</p>
6     <p>1111</p>
7     <p>1111</p>
8     <p>1111</p>
9     <p>1111</p>
10    <p>1111</p>
11    <p>1111</p>
12    <p>1111</p>
13    <mt-button type="primary" @click="simple">简捷提示框</mt-button>
14    <mt-button type="primary" @click="normal">标准提示框</mt-button>
15    <mt-button type="primary" @click="confirm">确认提示框</mt-button>
16    <mt-button type="primary" @click="prompt">输入提示框</mt-button>
17    <p>1111</p>
18    <p>1111</p>
19    <p>1111</p>
20    <p>1111</p>
21    <p>1111</p>
22    <p>1111</p>
23    <p>1111</p>
24    <p>1111</p>
25    <p>1111</p>
26    <p>1111</p>
27    <p>1111</p>
28    <p>1111</p>
29    <p>1111</p>
30    <p>1111</p>
31    <p>1111</p>
32    <p>1111</p>
33    <p>1111</p>
34  </div>
35 </template>
36 <script>
37 export default {
38   methods:{
39     prompt(){
40       this.$messagebox.prompt('请输入QQ群名称:', '新建QQ群', {
41         inputType: 'text',
42         inputPlaceholder: 'QQ群名称'
43       }).then(action=>{
44         console.log(action);
45       }).catch(err=>{
46
47       });
48     },
49     simple(){
50       this.$messagebox('标题', '提示内容');
51     },
52     normal(){
53       this.$messagebox.alert('提示内容', '标题');
54     },
55     confirm(){
56       this.$messagebox.confirm('提示内容', '标题', {
57         confirmButtonText: 'OK',
58         confirmButtonHighlight: true
```

```

58     confirmButtonHighLight:true,
59     cancelButtonText: 'Cancel',
60     cancelButtonHighLight:true,
61     //是否在单击遮罩层时关闭对话框
62     closeOnClickModal:false,
63     //是否允许通过键盘ESC按钮关闭对话框
64     closeOnPressEscape:false,
65     //是否锁定滚动条
66     lockScroll:true
67   }).then(action=>{
68     console.log(action);
69   }).catch(err=>{
70
71   });
72 }
73 }
74 }
75 </script>
76 <style>
77 .mint-msgbox-confirm{
78   color: #F00 !important;
79 }
80 .mint-msgbox-confirm:active {
81
82   font-weight: bold;
83 }
84
85 .mint-msgbox-cancel:active {
86   color: #0563a9;
87   font-weight: bold;
88 }
89 </style>

```

## 4.作业

---

- 修改 src/views/Register.vue 组件文件，如果用户名为 admin 且密码为 admin 时弹出提示框，内容为 对不起，该用户已存在，否则询问 用户注册成功，是否立即登录？，如果选择 是 时，跳转到登录的路由，如果选择 否，则跳转到首页路由
- 复习MySQL的 CURD 操作
- 复习 MySQL连接池 的相关知识