1. **项目文件介绍**
2. S70前后端分离版本，数据层沿用MVC版本的，可以MVC版本共用
3. 沿用MVC版本的下拉组件，用法与MVC版本一致：

字典、系统参数、菜单、组织结构、角色、权限

1. 该系统建表要求，每个表都需要包含6个字段：

|  |  |  |
| --- | --- | --- |
| 字段名称 | 类型 | 备注 |
| Creator | int | 创建人ID |
| CreateDate | datetime | 创建时间 |
| Updater | int | 更新人ID |
| UpdateDate | datetime | 更新时间 |
| IsDel | tinyint（bool） | 是否删除 |
| CompanyID | int | 商户ID |

1. 目录说明



* Toowell.S70.View：分离版本前端
* Toowell.S70.Api：分离版本后端API
* Toowell.S70.Apis：API引用的子项目，类似MVC版本
* Toowell.Common：通用类
* Toowell.S70.Generator：代码生成器
* Toowell.S70.Migrations：数据表生成器

1. **前端（Toowell.S70.View）**
   1. 前端结构

├─.env.development --测试环境配置

├─.env.production --生产环境配置

├─public --编译后，可以通过该文件修改，图片和配置

│ ├─img --图片

│ └─config.js --配置（包含参数和API地址）

└─src

├─api --API接口

├─assets --静态文件

├─components --组件（包含MVC版本的下拉组件）

├─config --参数配置

├─directives --常用类

├─layout --布局

├─locales --本地化

├─router --路由

├─store --状态保持

├─style --样式

├─utils --通用类

└─views --vue视图

├─Company --多商户管理

├─dashboard -- 工作台

├─Dictionary -- 字典

├─login -- 登录界面

├─LogonLog -- 登录日志

├─Menu -- 菜单

├─NtService -- 通知列表（测试）

├─Operation -- 权限列表

├─Organization -- 部门列表

├─Role --角色列表

├─User --用户列表

├─userCenter --用户中心

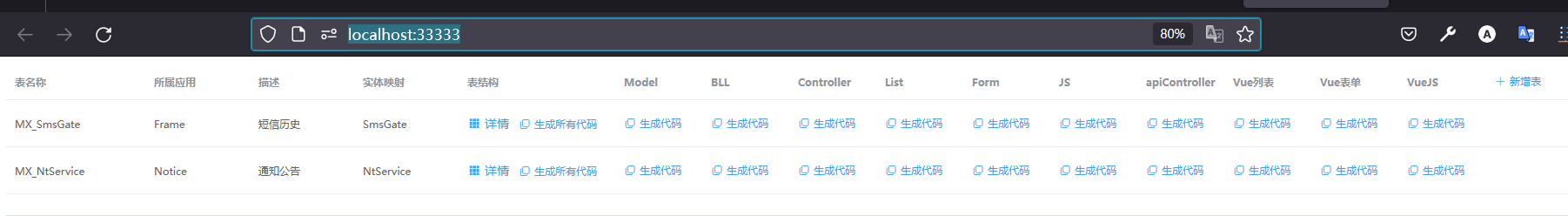
└─UserOnline --在线列表

1. 运行与发布

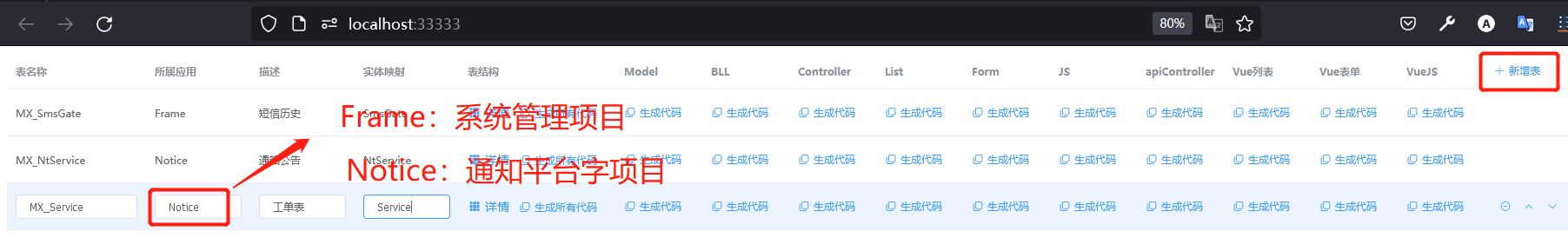
* 安装依赖运行：npm install
* 打开站点运行：npm run serve
* 发布站点运行：npm run build
* 发布完，在根目录下生成dist文件夹
* 可以把dist文件夹单独发布一个前端
* 也可以把dist文件夹的内容复制到后端wwwroor文件夹与webapi一起发布

1. **代码生成器（Toowell.S70.Migrations）**

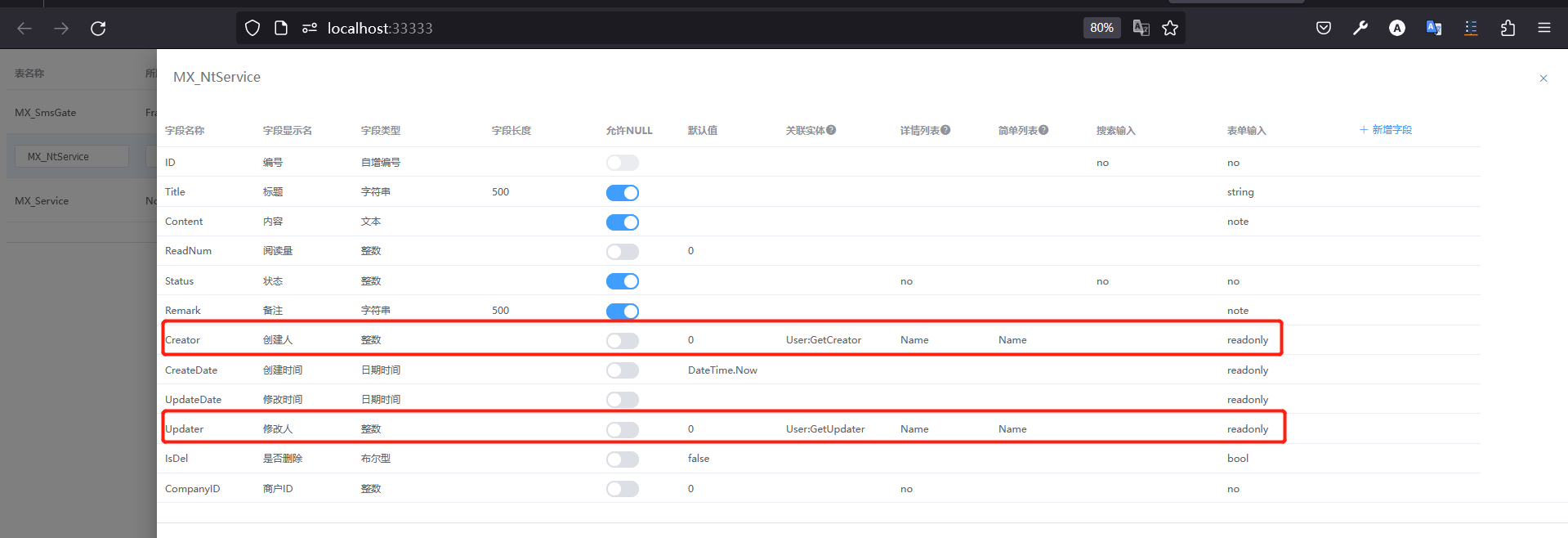
* 如果有修改里面的代码，执行Toowell.S70.Generator\Generator.bat
* 如果没有修改，执行Toowell.S70.Generator\Toowell.S70.Generator.exe
* 自动打开界面：http://localhost:33333/



* 右上角新增表：



* 点击列表【详情】，新增字段



* 其中类型，Creator创建人ID，需要关联其他表的，请参考MX\_NtService的设置方法
* 设置好后，在列表点击【生成所有代码】生成所有的代码
* 或者分别点击【生成代码】生成对应的代码

1. Vue前端

* Toowell.S70.View\src\api\NtService.js 前端api接口
* Toowell.S70.View\src\views\NtService\List.vue 列表文件
* Toowell.S70.View\src\views\NtService\Form.vue 表单文件

1. WebApi后端

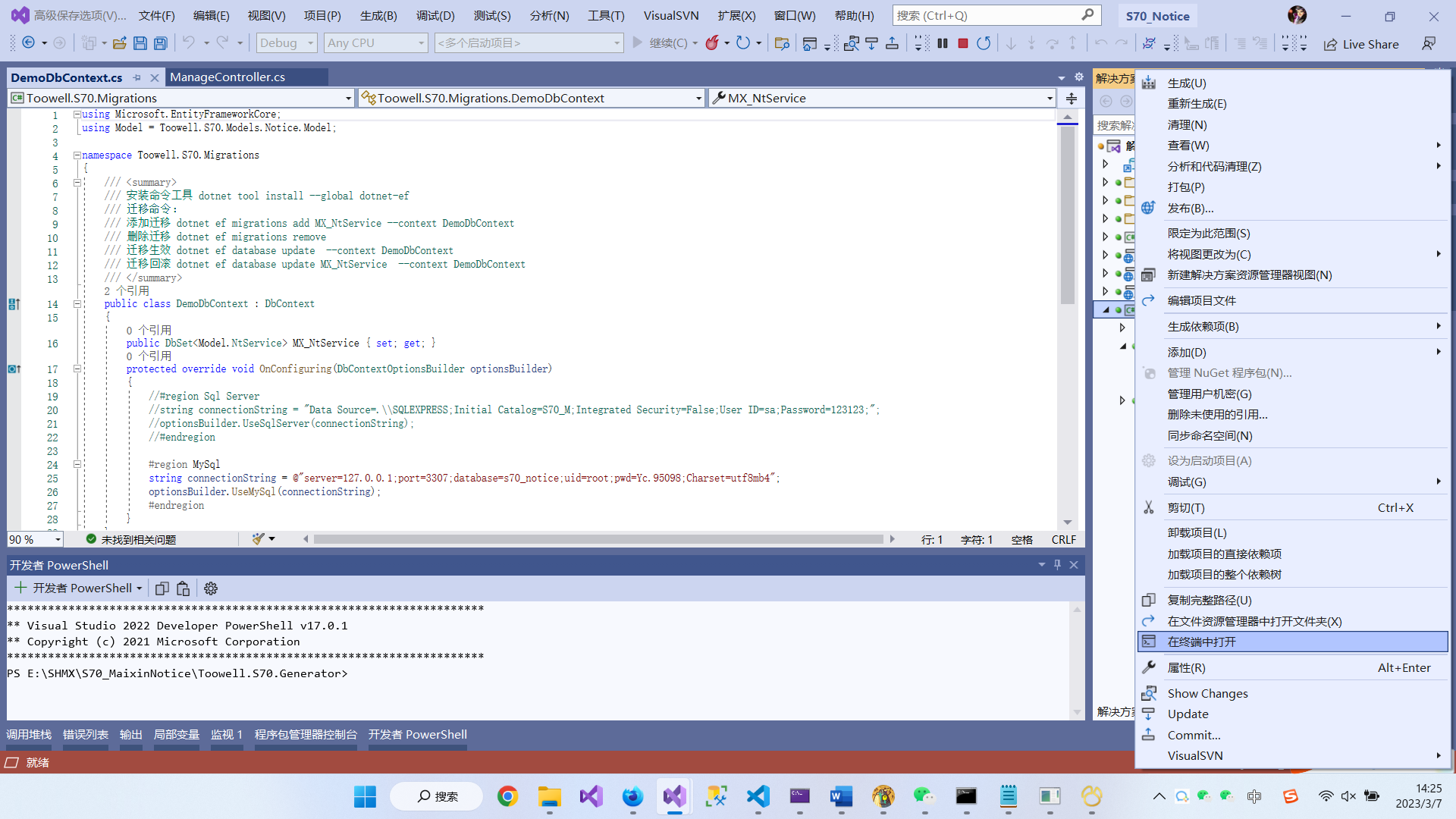
* Toowell.S70.Apis\Toowell.S70.Apis.Notice\Controller\NtServiceController.cs 接口

1. Model层

* Toowell.S70.Models\Toowell.S70.Models.Notice\Model\NtService.cs 实体
* Toowell.S70.Models\Toowell.S70.Models.Notice\Model\Extension\NtService.cs扩展
* Toowell.S70.Models\Toowell.S70.Models.Notice\BLL\NtService.cs 业务

1. **数据库生成器（Toowell.S70.Migrations）**

* 右键【Toowell.S70.Migrations】项目打开终端：

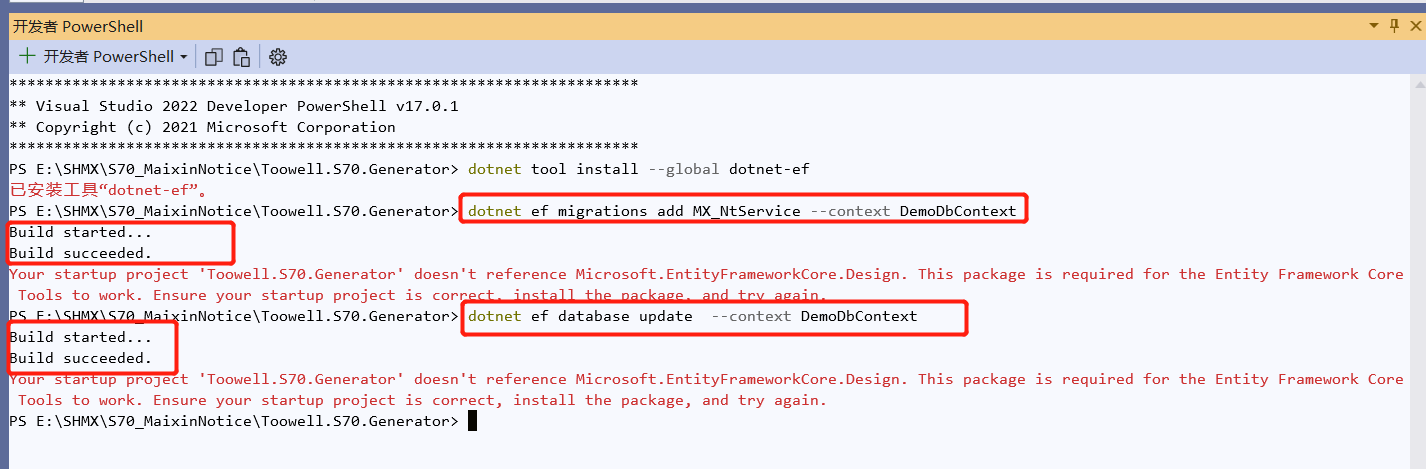


* 在【DemoDbContext.cs】设置好数据库链接后
* 在终端分别输入：（MX\_NtService为数据表名）

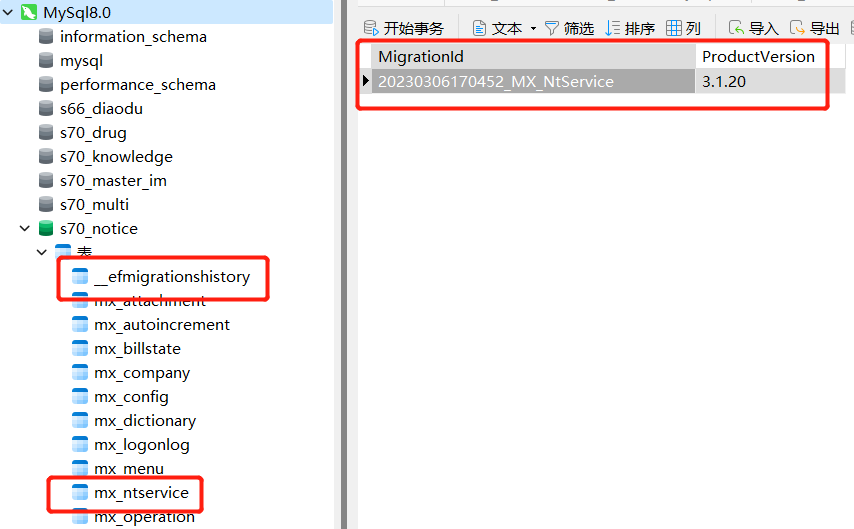
安装命令工具 dotnet tool install --global dotnet-ef

添加迁移 dotnet ef migrations add MX\_NtService --context DemoDbContext

迁移生效 dotnet ef database update --context DemoDbContext



* 执行完成后，可以再数据库看到生成的表



1. **前端组件使用**

**前端参考框架：【SCUI Admin】地址：https://lolicode.gitee.io/scui-doc/**

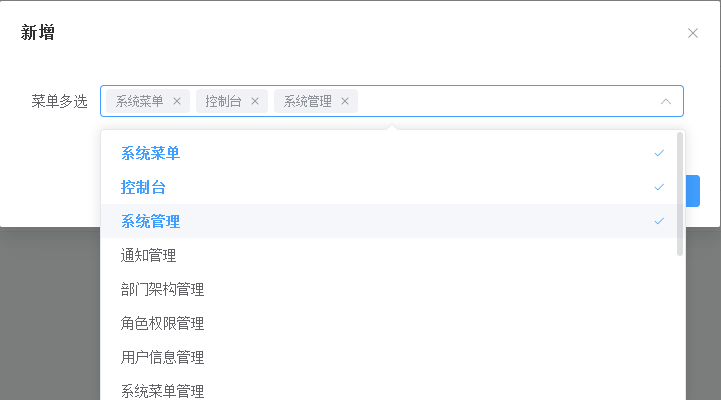
* 1. 下拉、级联、文本框，一些比较常用的属性
* 文本占位：placeholder
* 禁用：disabled
* 搜索：filterable
* 清空：clearable
* 下拉、级联多选：multiple
  1. 官方文档
* 级联：<https://element-plus.gitee.io/zh-CN/component/cascader.html>
* 下拉：<https://element-plus.gitee.io/zh-CN/component/select.html>
  1. 例子
* 菜单级联下拉多选

<mx-menu-cascader v-model="form.ID" clearable filterable multiple></mx-menu-cascader>



* 菜单下拉多选

<mx-menu-select v-model="form.ID" clearable filterable multiple></mx-menu-select>



* 字典下拉多选

dictType：字典类型

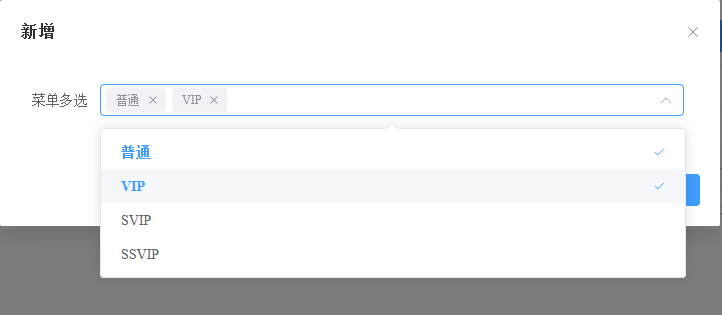
valueKey：值类字段

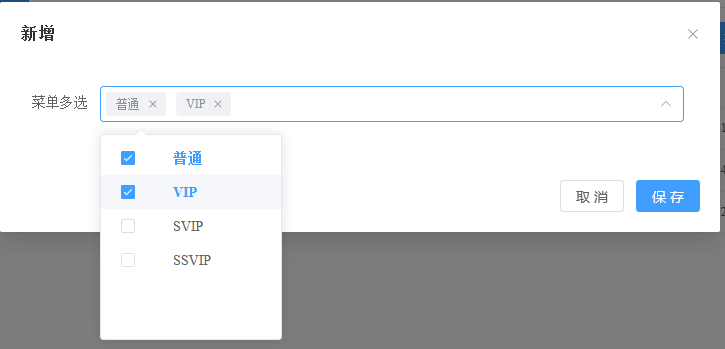
parentId：父级ID

<mx-dict-select v-model="form.ID" dictType="CompanyLevel" valueKey="Code" placeholder="字典" clearable filterable multiple></mx-dict-select>

* 或者

<mx-conf-input v-model="form.ID" input-type="dict@CompanyLevel" placeholder="企业等级" multiple></mx-conf-input>





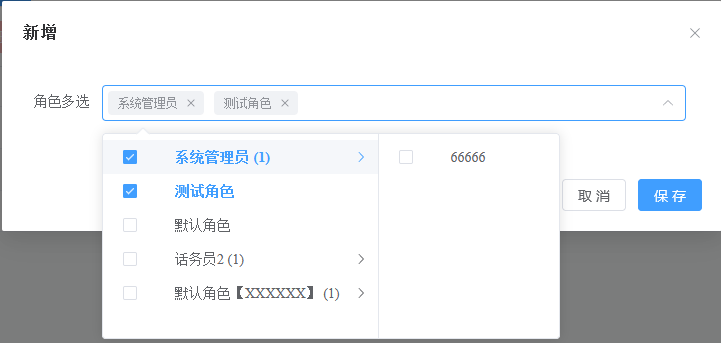
* 组织结构级联多选

<mx-org-cascader v-model="form.ID" multiple placeholder="组织结构"></mx-org-cascader>



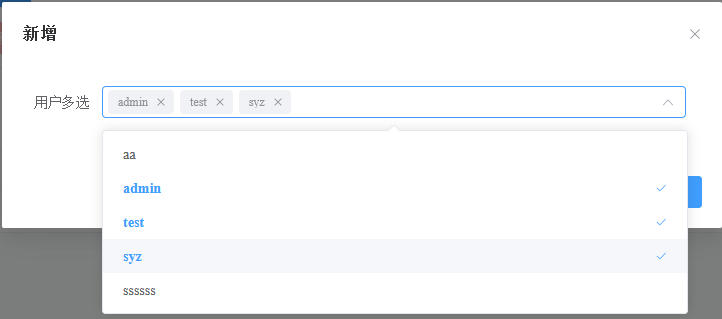
* 角色级联多选

<mx-role-cascader v-model="form.ID" multiple placeholder="角色多选"></mx-role-cascader>



* 用户下拉多选

<mx-user-select v-model="form.ID" multiple placeholder="用户多选"></mx-user-select>



* 省市县级联

<mx-conf-input v-model:province="form.Province" v-model:city="form.City" v-model:area="form.Area" input-type="region" placeholder="省市县" clearable></mx-conf-input>



* 自定义下拉
* apiObj为api接口，返回格式为数组，可以自定显示

<sc-select v-model="form.ID" :apiObj="$API.system.user.list" clearable filterable style="width: 100%;">

    <template #option="{data}">

        <span style="float: left">{{ data.UserName }}</span>

        <span style="float: right; color: #999; font-size: 13px">{{ data.ID }}</span>

    </template>

</sc-select>

