# Jeecg-Boot平台开发手册

> date:2020-09-14
>
> author:甘树禧（Succy）,黄国才

**声明**：本平台基于Jeecg-Boot开源平台扩展，如有任何问题请先查看官方WIKI地址

**思维导图如下**：


# 环境搭建

## 开发工具安装

> Git仓库安装包下载： 点击下载
>
> 其余安装包下载地址：链接：https://pan.baidu.com/s/16z9qNtyk24bsrZxRFBHP2w 提取码：pagv
>
> 环境搭建参考文档：官方教程
>
> 环境搭建视频链接：点击下载

## 工具清单:

- Git-2.11.0-64-bit.exe
- node-v12.13.1-x64.msi
- yarn-1.21.1.msi
- ideaIU-2019.2.3.exe （前后端开发工具均推荐使用IDEA，如使用其他工具，请自行参照官方文档配置)
- jdk-8u191-windows-x64.exe
- apache-maven-3.5.4.zip
- redis64-3.0.501 （开发可统一使用开发环境Redis服务，不需自行搭建redis服务)
- mysql-5.7.26-winx64.zip （开发可统一使用开发库，不需自行搭建数据库环境)

## 前端环境搭建

- 安装Node.js
- 安装yarn

- git克隆远程仓库

  `git clone http://172.16.4.191:3000/JEECG/ant-design-vue-jeecg.git`
- 使用IDEA导入项目
- 通过yarn install安装项目依赖
- 通过yarn run serve启动项目

### 后端环境搭建

- JDK1.8安装
- Maven安装并配置柳钢私服仓库地址

```
<mirror>
    <id>lg-nexus</id>
    <mirrorOf>*</mirrorOf>
    <name>Nexus of lg</name>
    <url>http://172.16.4.191:8081/repository/maven-public/</url>
</mirror>
```

- Redis安装　（使用统一的开发Redis无需安装该环境）
- git克隆后台远程代码仓库

  `git clone http://172.16.4.191:3000/JEECG/jeecg-boot.git`
- 使用IDEA打开项目
- 直接运行Application主类（基于SpringBoot方式启动,请先确定数据库与Redis连接的正确性）

## Online表单开发

Online表单开发是Jeecg框架核心功能，Jeecg的核心思想就是低代码开发，亦称零代码开发。

学习视频下载链接： 点击下载

## 代码生成器

当业务定制化内容较多，Online表单开发满足不了生产需求亦或是需要代码集成发布时，就需要用到代码生成器。

学习视频下载链接：点击下载

## 权限管理

Jeecg的权限管理基于shiro+JWT实现。

学习视频下载链接：点击下载

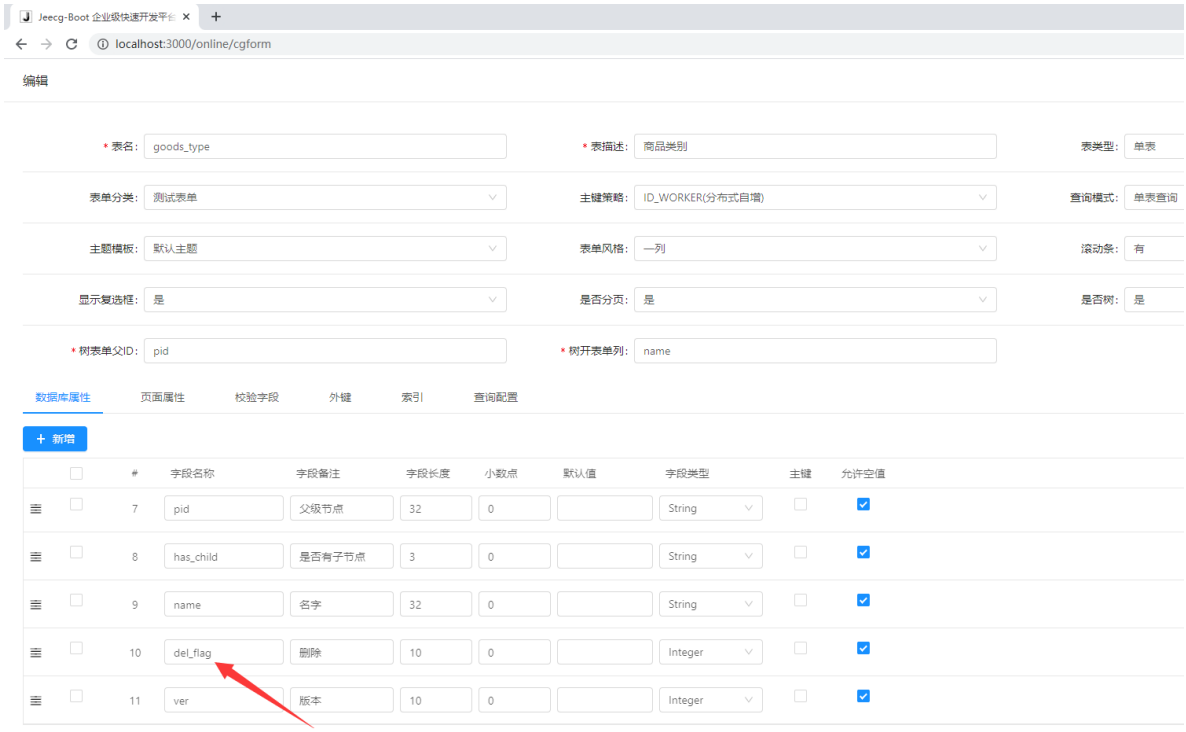
## 框架扩展

# 逻辑删除

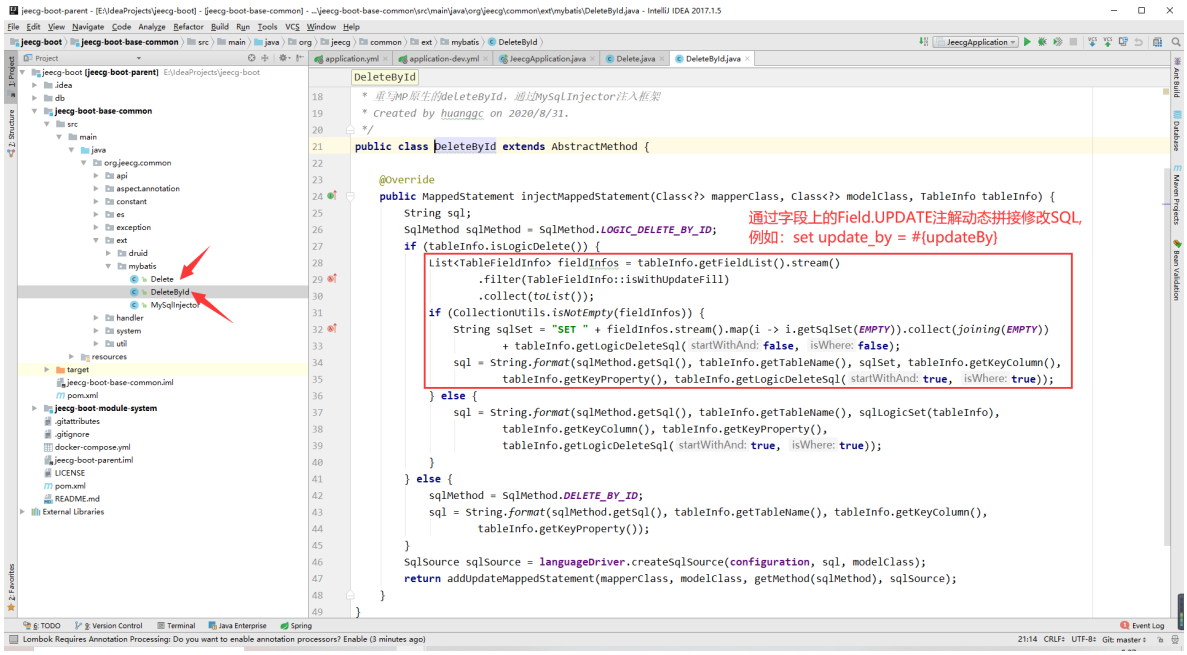
基于Jeecg-Boot原框架，我们拓展了自己的逻辑删除业务。官方的逻辑删除不会联动设置update*字段，导致数据删除无法追溯，基于该业务场景，我们拓展了自己的逻辑删除实现。

**特别注意：逻辑删除在online在线表单功能测试并不会生效，只有在代码生成之后才会生效。**

**使用方法：在online在线表单的字段增加del_flag即可。**如下图所示

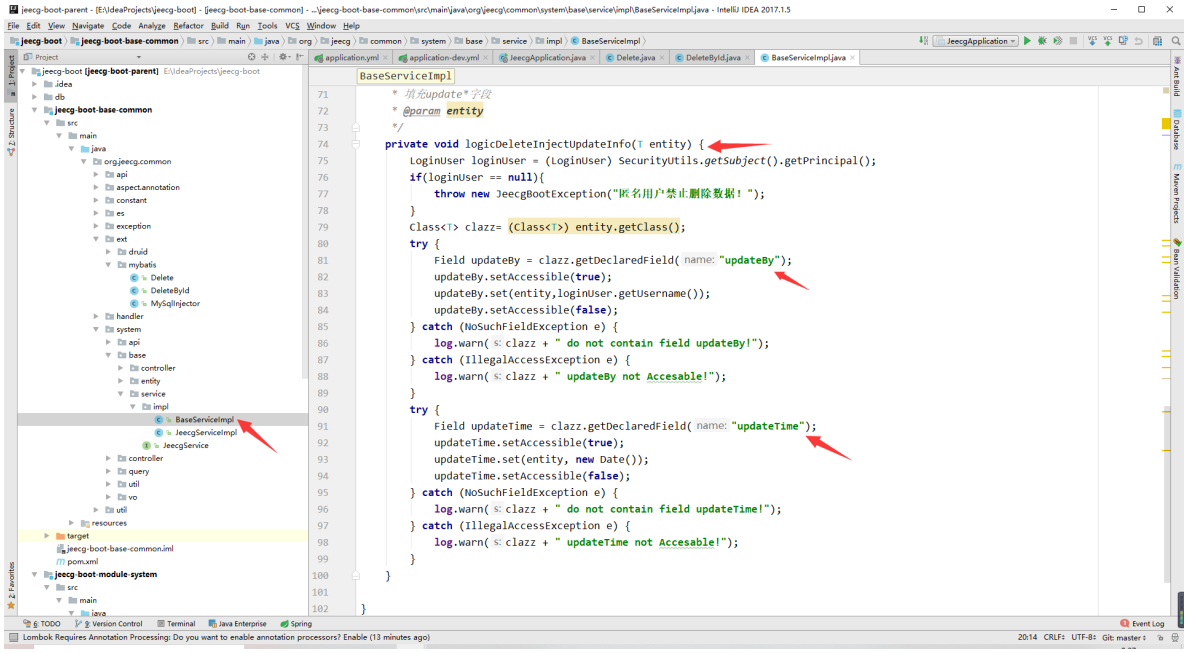



**实现原理：**通过覆盖Mybatis的删除方法Delete（条件删除）和DeleteById（Id删除，**推荐**）,并在BaseService里面通过反射操作实体类对象设置update*字 段,完成数据联动效果，重要代码如图示：



```java
/**更新人*/
@TableField(fill = FieldFill.UPDATE)
@ApiModelProperty(value = "更新人")
private java.lang.String updateBy;
/**更新日期*/
@JsonFormat(timezone = "GMT+8",pattern = "yyyy-MM-dd HH:mm:ss")
@DateTimeFormat(pattern="yyyy-MM-dd HH:mm:ss")
@TableField(fill = FieldFill.UPDATE)
@ApiModelProperty(value = "更新日期")
private java.util.Date updateTime;
/**所属部门*/
```

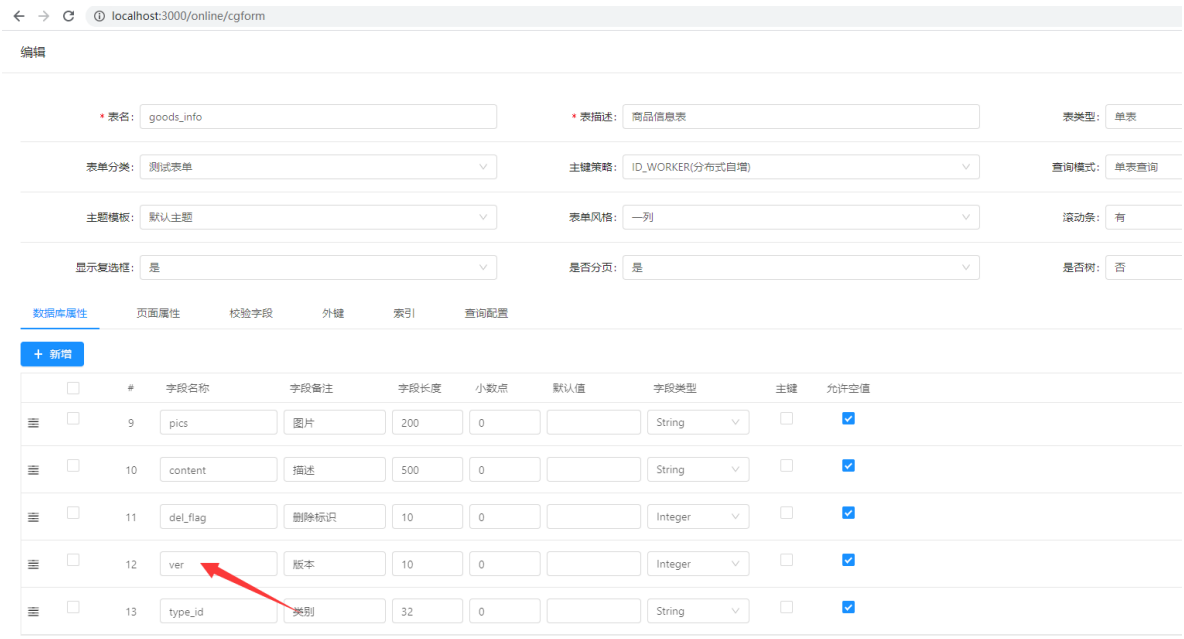
实体类update*上加上该注解
供DeleteById方法联动使用




# 乐观锁

为防止用户并发操作数据修改导致数据异常，加入乐观锁版本校验机制（**按需使用**），实现方法参照 [Mybatis-Plus官方文档]

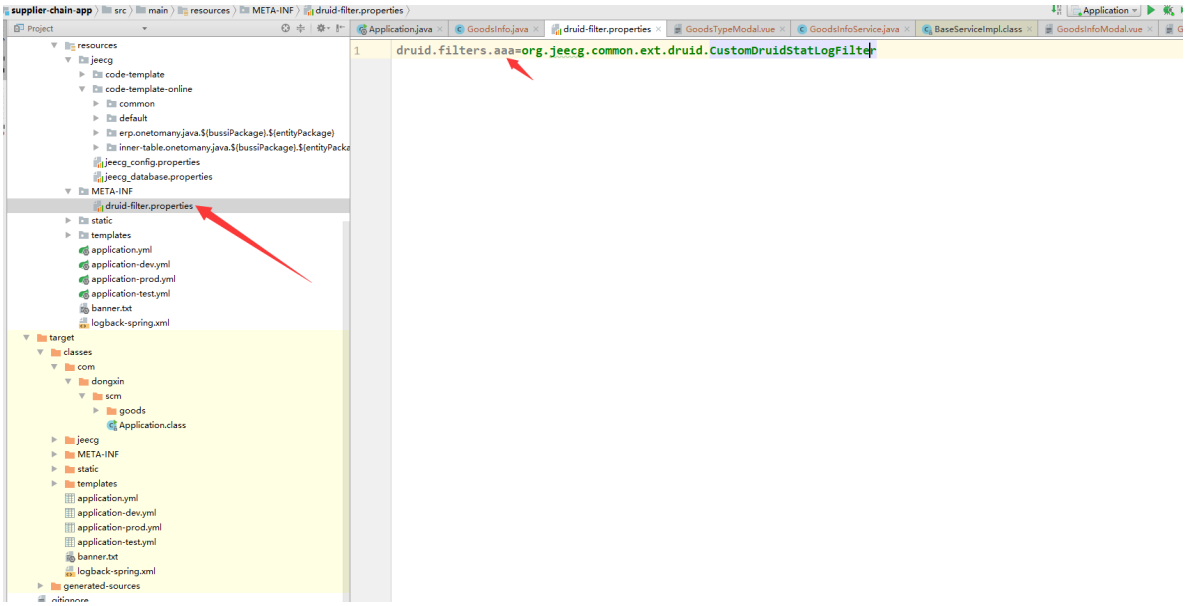
**使用方法：在online表单字段中加上ver字段即可**，如下图示：

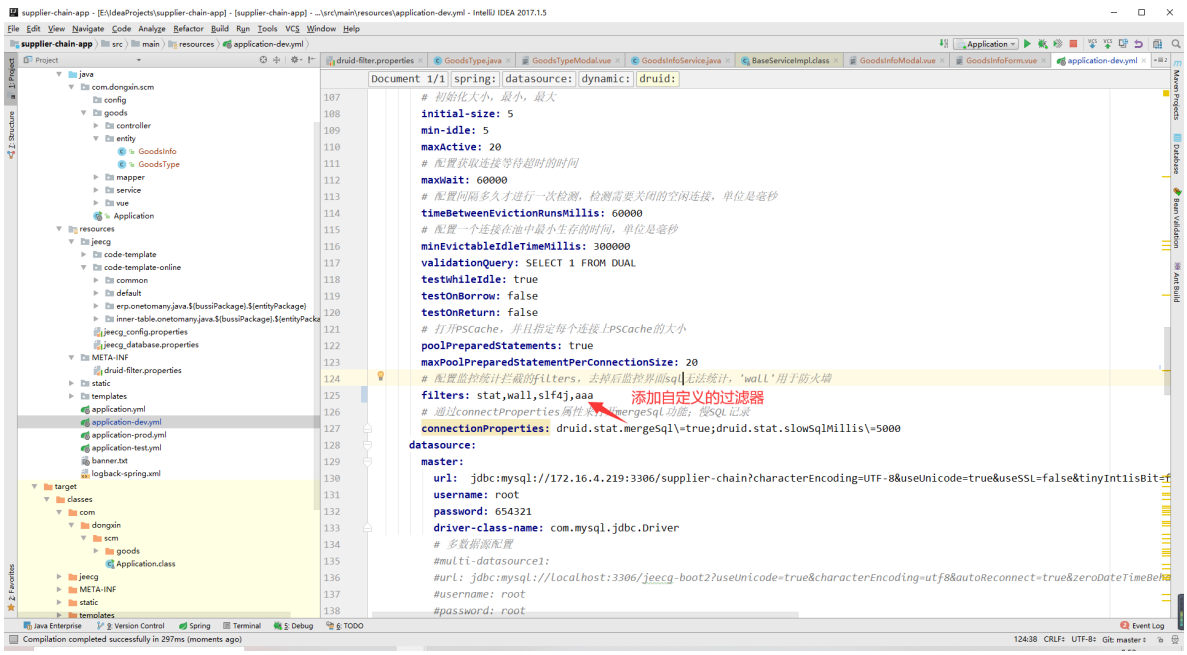



**注：代码模板会通过ver字段在实体类智能加上@Version注解。**

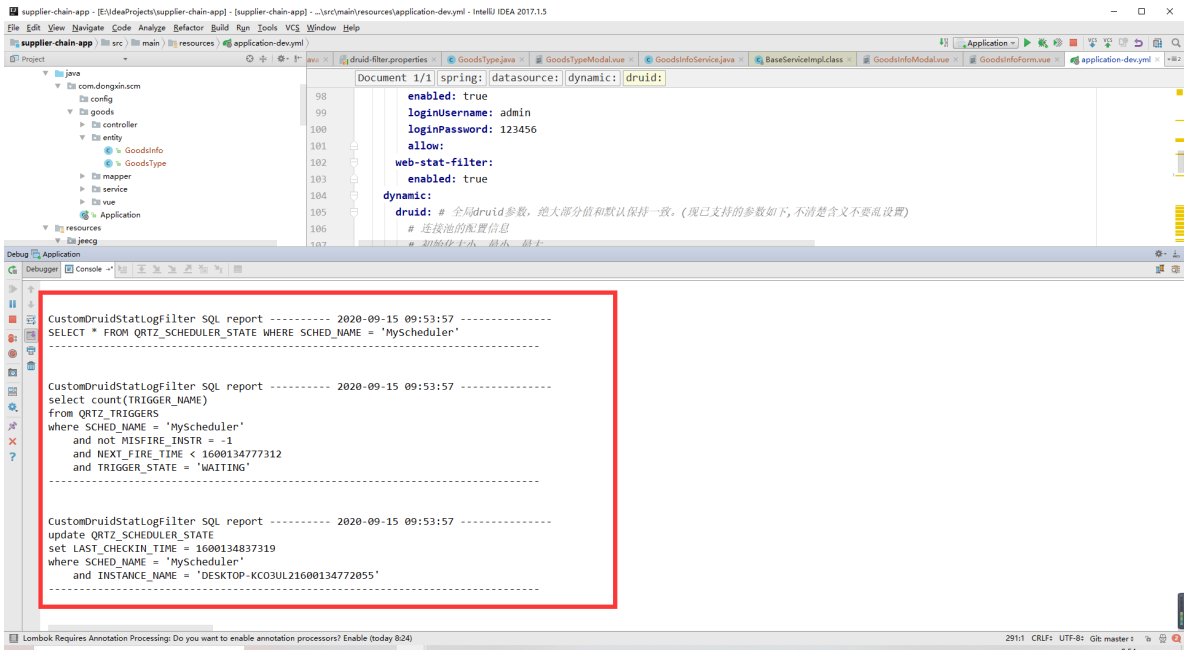
# 自定义DruidFilter打印sql日志

通过SPI机制注入自定义的扩展过滤器，实现sql日志更清楚的打印到控制台






**现象：**




# 后端项目骨架生成

基于不用的业务项目需要使用Jeecg-Boot框架开发，可直接使用maven骨架生成后台项目模板。

```
mvn archetype:generate -B ^
    -DarchetypeGroupId=com.dongxin ^
    -DarchetypeArtifactId=jeecgframework-archetype ^
    -DarchetypeVersion=RELEASE ^
    -DgroupId=com.dongxin ^
    -DartifactId=demo ^
    -Dversion=1.0-SNAPSHOT
```

学习视频下载链接：[点击下载]

# 前端开发小技巧

## 前端脚手架基本结构认识

- main.js　入口文件
- router　动态加载路由详见官方文档
- store　缓存
- permission　权限验证

## 开发常见问题：

### 异步加载树

**问题背景**：柳钢集团部门结构复杂，公司部门量级在四位数，一次加载到页面dom元素导致页面加载缓慢，如何解决？

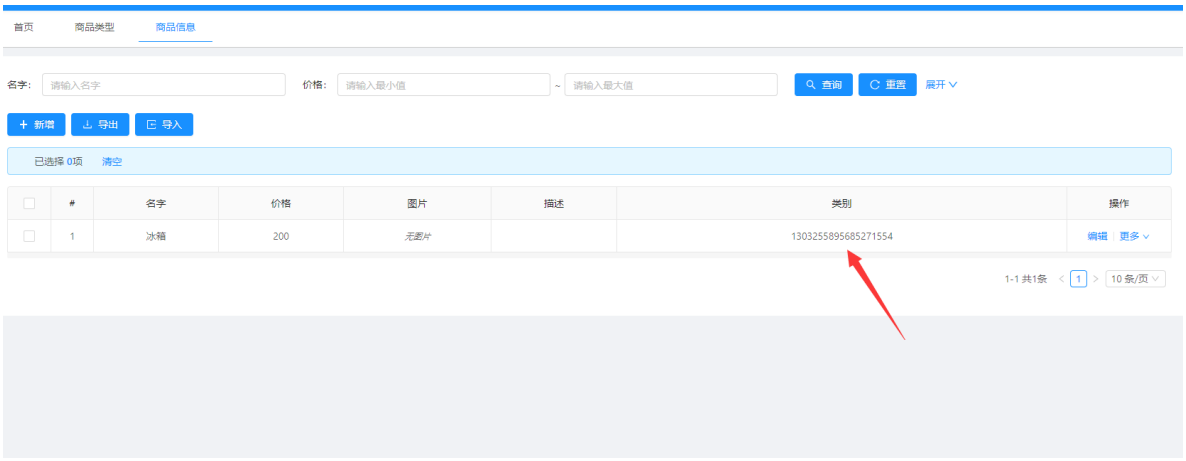
**解决方案**：异步加载dom树节点，默认关闭子节点，参考antdv官方文档之异步加载树

```
<a-tree
  checkable
  multiple
  @select="onSelect"
  @check="onCheck"
  @rightClick="rightHandle"
  :selectedKeys="selectedKeys"
  :checkedKeys="checkedKeys"
  :treeData="departTree"
  :checkStrictly="checkStrictly"
  :expandedKeys="iExpandedKeys"
  :autoExpandParent="autoExpandParent"
  :loadData="asyncLoadTree"
  @expand="onExpand"/>
```

```
asyncLoadTree(treeNode){
    return new Promise(resolve => {
      if (treeNode.dataRef.children) {
        resolve();
        return;
      }
      console.log(treeNode.dataRef.title + "节点异常")
    });
},
```

### 前端页面关联类别文本渲染

**问题背景**：通过online表单代码生成后的功能列表显示会直接显示存储id或code，期望展示的是对应关联表文本。




**解决方案**：在字段上使用前端渲染，如下图所示。

```
},
{
  title:'类别',
  align:"center",
  dataIndex: 'typeId',
  customRender: (text) => {
      return this.filterGoodsType(text);
  }
},
{
```

```
created() {
  queryGoodsType(null).then((res)=>{
    if(res.success){
      this.goodsTypeOptions = res.result.records;
    }else{
      console.error('queryGoodsType error: : ', res.message)
    }
  })
},
computed: {
  importExcelUrl: function(){
    return `${window._CONFIG['domianURL']}/${this.url.importExcelUrl}`;
  },
},
methods: {
  initDictConfig(){
  },
  filterGoodsType(id){
    for (let goodsType of this.goodsTypeOptions) {
        if (id.toString() === goodsType.id.toString()) {
          return goodsType.name.toString();
        }
    }
    return id;
  }
}
```

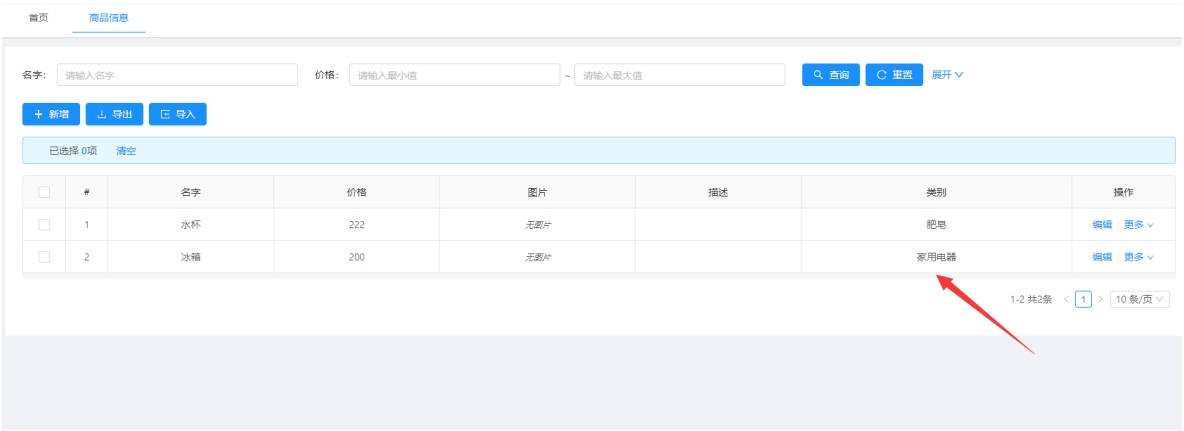
钩子方法请求后台完成属性初始化

声明渲染方法

在api定义如下接口并导出

```
const queryGoodsType = (params)=>getAction("/goods/goodsType/childList",params);
```

最终效果如下图：




## 某些组件（例如自定义树）在代码生成后控件发生变化

**问题背景：**商品类型表为树，商品信息表的类别字段引用了该树，online表单功能正常，在生成代码后组件变成文本输入框。

**解决方法：**参考商品类型表单的树结构，直接复制该控件并修改对应属性。