6 ReadWriteAuthorization 组件

所在程序集：Phenix.Services.Client

命名空间：Phenix.Services.Client.Security

组件功能：控制 WinForm 界面上控件所绑定数据的读写授权，涉及其属性：ReadOnly（DEV 控件：Properties.ReadOnly）

6.1 前提条件

事先需通过 Phenix.Services.Host.exe 注册业务组件，具体见“05.业务对象公共接口的授权”章节。

6.2 简介

ReadWriteAuthorization 组件属于界面控制层。

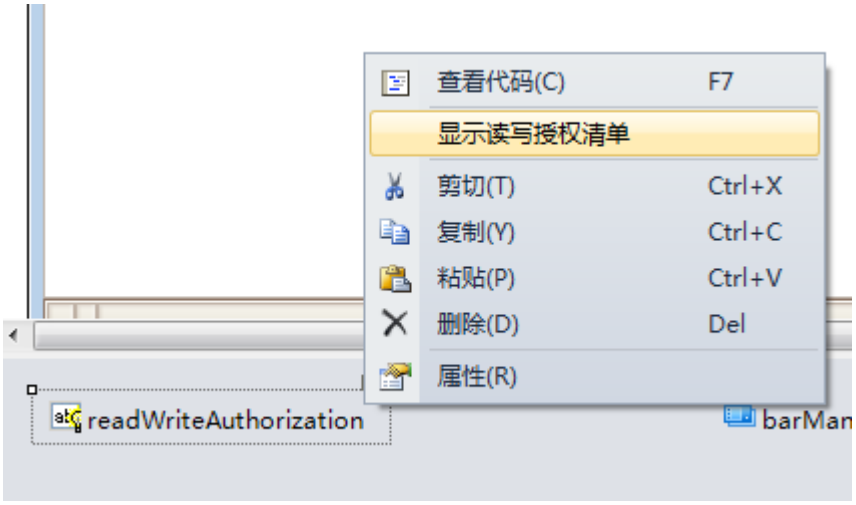
将业务对象属性的读写授权体现在 UI 界面上，可以有效引导操作者，并预防操作者的误操作，用户体验非常友好。

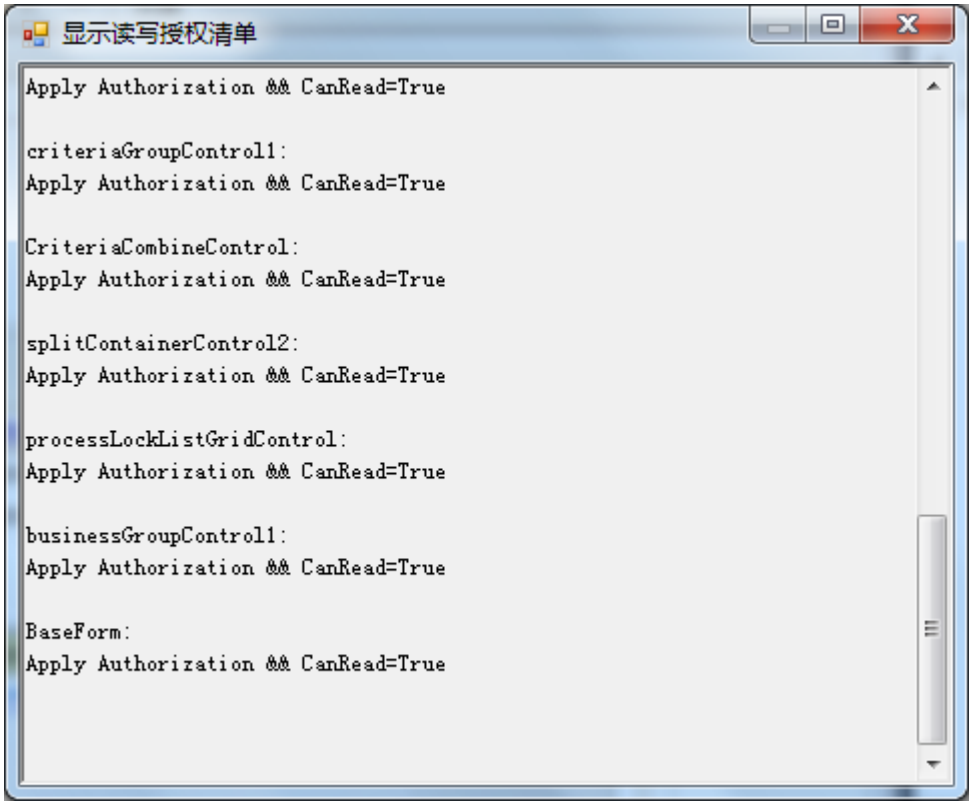
本组件改写自 CSLA 的 ReadWriteAuthorization 组件，改写的缘由是能够让它支持到 Developer Express 控件包。

6.3 配置

通常，业务数据的编辑界面都会使用到 Phenix.Windows.BarManager 组件，所以为了简化界面设计步骤，它封装了 ReadWriteAuthorization 组件，自动处理界面数据的读写授权，一般情况下这些界面无需再添加 ReadWriteAuthorization 组件。

ReadWriteAuthorization 组件是通过将属性扩展到容器中控件的方法来实现配置的，如果要查看整体配置情况的话，可以：

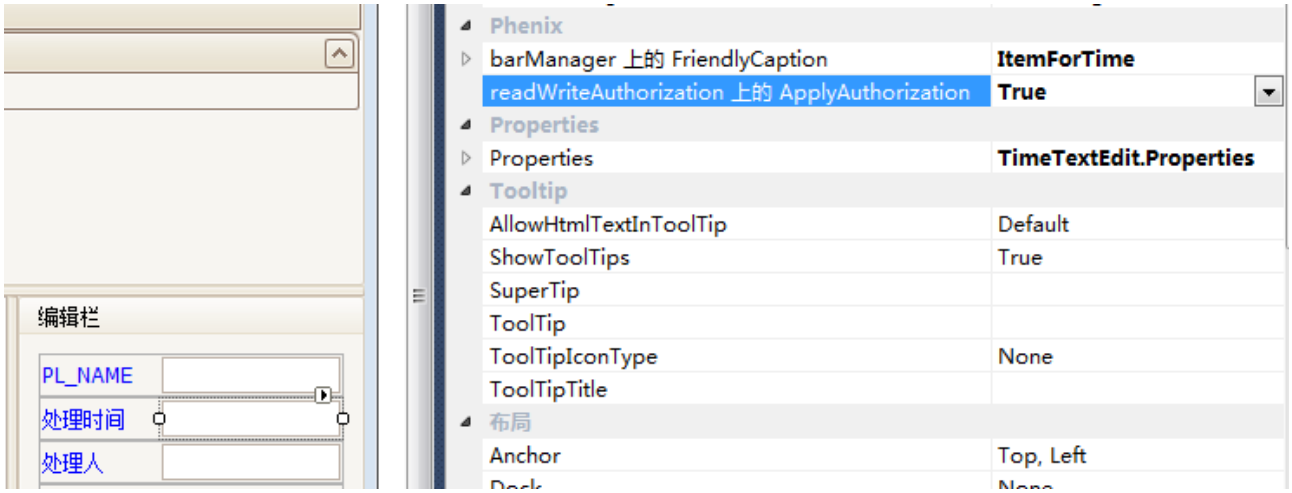




下面是供配置的属性：

属性	说明
ApplyAuthorization	是否应用授权：当为 true 时，被扩展控件将受到本组件的读写授权管控

设计界面：



将一些不相关控件的扩展属性 ApplyAuthorization 设置为 false 后，应用系统的运行性能会有些微的提升。

6.4 操作

封装了 ReadWriteAuthorization 组件的 Phenix.Windows.BarManager 会根据以下界面中所操作数据的变化而自动控制它们的读写授权：

- 界面初始化后；
- 切换了当前操作的 BindingSource；
- 当前操作的 BindingSource 的操作状态（浏览、编辑）发生了切换；
- 当前操作的 BindingSource 的 DataSource、List、Item、Position 发生了改变；

如果你的界面上没有 Phenix.Windows.BarManager 组件的话，请按照上述条件在相关组件的事件中自行调用 ReadWriteAuthorization 组件的接口函数：

```
/// <summary>
/// 重置控件的读写授权
/// </summary>
/// <param name="readOnly">只读</param>
/// <param name="sources">数据源队列</param>
public void ResetControlAuthorizationRules(bool readOnly, params BindingSource[] sources)
```

下述代码是将界面上所有扩展属性 ApplyAuthorization = true 的控件设置为只读：

```
this.readWriteAuthorization.ResetControlAuthorizationRules(true);
```

也可以有针对性地执行读写授权，比如指定具体某些 BindingSource 下的控件为只读：

```
this.readWriteAuthorization.ResetControlAuthorizationRules(true,
this.processLockListBindingSource);
```

6.5 表现

ResetControlAuthorizationRules 函数的 readOnly 参数传入的是你希望达到什么样的效果，实际结果要受到权限配置、业务规则的影响，如果其中有一项的验证结果是“不允许写入”，结果就只能是 readOnly = true。