## 客户端僵死问题（deshuai.kong）

google 查了一下，该问题可能和<http://stackoverflow.com/questions/559241/need-help-deciphering-a-c-sharp-stack-trace>这个问题是一样的，是由于在其它线程中执行了GUI中代码，微软推荐的解决方案中是使用InvokeRequired+代理来解决，但实际上在2.0之后可能这个InvokeRequired 会有一些问题（<http://ikriv.com/dev/dotnet/MysteriousHang.html>），

目前解决：

1.涉及有动态创建控件的要实例完，在调用实例的Handle属性  
如：  
RateModifyTable rt = SpringHelper.GetObject<RateModifyTable>("RateModifyTable");  
rt.Handle

日志排查:

1.在Program 中将主线程设置一个名字  
void Main()

{ Thread.Current.Name = "MainGUI"; ... }

2.提供一个公共的方法，放到Atf.Common中  
void EnsureMainGuiThread()  
{  
if (Thread.Current.Name != "MainGUI")

{ LogHelper.Error(Thread.Current.Name+" This code must execute on the main GUI thread"); }

}  
3.在所有使用代理的地方,进行检查  
void SomeMethod()  
{  
if (InvokeRequired)

{ BeginInvoke(...); }

else

{ EnsureMainGuiThread(); do() }

...  
}

4.在Program 中Mian 中增加一个事件  
SystemEvents.UserPreferenceChanged += new UserPreferenceChangedEventHandler(SystemEvents\_UserPreferenceChanged);

static void SystemEvents\_UserPreferenceChanged(object sender, UserPreferenceChangedEventArgs e)  
{  
LogHelper.Debug(string.Format("sender:

{0}

,e:

{1}

",sender,e.Category));  
}

测试程序：

## GridControl DataTable 配合使用(deshuai.kong)

分享一个小知识，请嘉琪和学文注意一下

目前我们开发的交易版面涉及有gridcontrol 要可以根据配置文件动态生成，同时有些列，可以通过配置计算公式或函数来实现自定义列，目前我们选用DataTable来作为

绑定GridControl 的数据源，有一些问题

1.如果处理不好，就会出现如下错误，界面就出现红叉

System.InvalidOperationException: DataTable 内部索引已损坏:“13”。

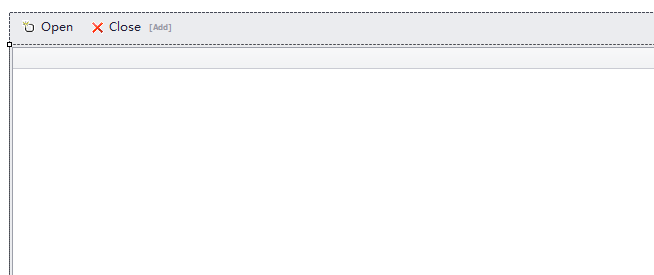
解决方法：在操作DataTable 之前，先通过DataTable.Copy（） 拷贝出来一个DataTable ,代码中有对DataTable 操作的，使用这个拷贝的DataTable 进行操作，操作完之后，将该拷贝的DataTable 重新赋值到GridControl 中。参考附件中的代码（官方提供的）

2. DataTable 支持表达式，参考<http://msdn.microsoft.com/zh-cn/library/system.data.datacolumn.expression(v=vs.80).aspx>

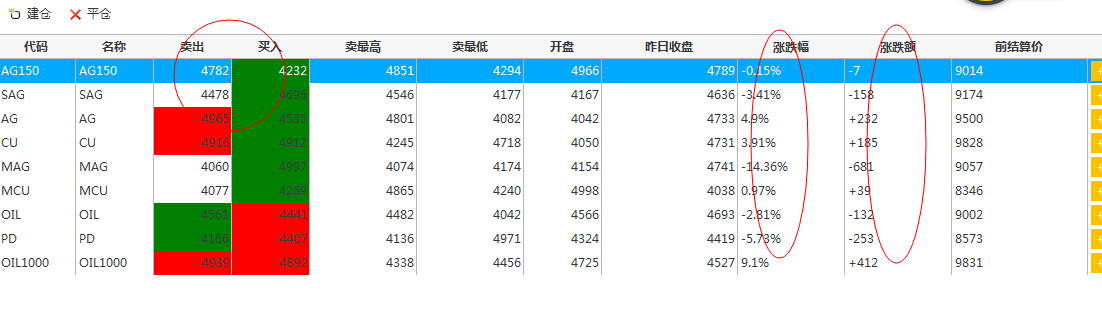
根据自己的需要编写对应的表达式

3.在报价浏览中测试可以满足需要

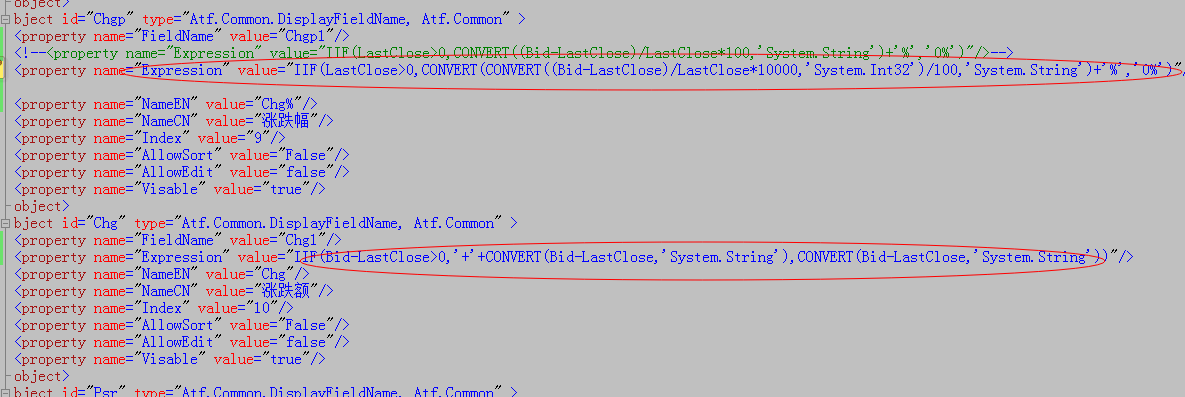
Vs 中



通过代码动态创建列



其中 涨跌幅，涨跌额是通过配置公式实现的



4，界面中的Bid,Ask 闪烁，也是通过配置达到

测试代码：

## Atf框架中，自定义控件，如何根据框架来改变皮肤和颜色（deshuai.kong）

public PositionsDetailsUI()

{

InitializeComponent();

ThemeAndMainColor.ThemeChanged += new EventHandler<StyleEventArgs>(ThemeAndMainColor\_ThemeChanged);

ThemeAndMainColor.MainColorChanged += new EventHandler<StyleEventArgs>(ThemeAndMainColor\_MainColorChanged);

this.Disposed += new EventHandler(positionsDetailsUI\_Disposed);

initParamter();

this.tableLayoutPanel2.ColumnStyles[2].Width = isEnglish.DisplayCaption.Equals("true") ? 55F : 35F;

ThemeAndMainColor\_ThemeChanged(null, null);

ThemeAndMainColor\_MainColorChanged(null, null);

}

#region 皮肤/颜色

void ThemeAndMainColor\_MainColorChanged(object sender, StyleEventArgs e)

{

string name = ThemeAndMainColor.MainColor;

Color c = PublicConsts.EngagingBaseColor;

if (name.ToLower() == "engaging")

{

c = PublicConsts.EngagingBaseColor;

}

if (name.ToLower() == "blushful")

{

c = PublicConsts.BlushfulBaseColor;

}

beClose.Buttons[0].Appearance.BackColor = beHedge.Buttons[0].Appearance.BackColor = beTP.Buttons[0].Appearance.BackColor = c;

}

void ThemeAndMainColor\_ThemeChanged(object sender, StyleEventArgs e)

{

switch (ThemeAndMainColor.Theme.ToLower())

{

case "black":

gridView1.Appearance.EvenRow.BackColor = Color.FromArgb(80, 80, 80);

gridView1.Appearance.OddRow.BackColor = Color.FromArgb(64, 64, 64);

gridView1.Appearance.HeaderPanel.BackColor = Color.FromArgb(64, 64, 64);

btnExport.Appearance.BackColor = Color.FromArgb(64, 64, 64);

this.tableLayoutPanel2.BackColor = Color.FromArgb(102, 102, 102);

break;

case "white":

gridView1.Appearance.EvenRow.BackColor = Color.Gainsboro;

gridView1.Appearance.OddRow.BackColor = Color.White;

tableLayoutPanel2.BackColor = Color.White;

break;

}

}

#endregion

## GridControl无数据时表格显示提示

//When no Records Are Being Displayed

private void gridView1\_CustomDrawEmptyForeground(object sender, CustomDrawEventArgs e)

{

//方法一（此方法为GridView设置了数据源绑定时，可用）

ColumnView columnView = sender as ColumnView;

BindingSource bindingSource = this.gridView1.DataSource as BindingSource;

if(bindingSource.Count == 0)

{

string str = "没有查询到你所想要的数据!";

Font f = new Font("宋体", 10, FontStyle.Bold);

Rectangle r = new Rectangle(e.Bounds.Top + 5, e.Bounds.Left + 5, e.Bounds.Right - 5, e.Bounds.Height - 5);

e.Graphics.DrawString(str, f, Brushes.Black, r);

}

//方法二（此方法为GridView没有设置数据源绑定时，使用，一般使用此种方法）

if (this.\_flag)

{

if (this.gridView1.RowCount == 0)

{

string str = "没有查询到你所想要的数据!";

Font f = new Font("宋体", 10, FontStyle.Bold);

Rectangle r = new Rectangle(e.Bounds.Left + 5, e.Bounds.Top + 5, e.Bounds.Width - 5, e.Bounds.Height - 5);

e.Graphics.DrawString(str, f, Brushes.Black, r);

}

}

}

## ComboboxEdit自定义组合控件(jiaqi.rao)

PopupContainerEdit + PopupContainerControl

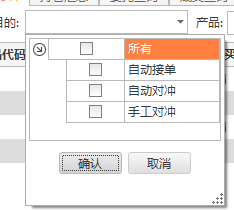
PopupContainerControl为画板，可以放任意控件，如：gridcontrol,treelist

PopupContainerEdit.properties.popupControl = PopupContainerControl即可完成组合

PopupContainerEdit.properties.ShowPopupCloseButton(是否显示关闭按钮)

GridControl里面使用同样对应控件即可。

效果图:



## Loading控件(deshuai.kong)

实现方式：自定义Loading控件a,需要Loading效果的控件b，

b.controls.add(a);

a.BringToFront();//将控件带到Z轴顺序前面

private Loading loading = new Loading();//不是线程安全的

protected void setUIStatus(bool status)

{

if (this.InvokeRequired)

{

this.Invoke(new Action<bool>(setUIStatus), new object[] { status });

}

else

{

this.toolbar.setBtnStatus(status);

this.pageControl.Enabled = status;

if (status)//加载完毕，隐藏Loading

{

loading.Hide();

}

else

{

loading.Show(gridControl1);//加载数据时候，显示Loading

}

}

}

protected override void search()

{

this.setUIStatus(false);//加载数据，禁用界面，同时显示Loading

ThreadPoolHelper.QueueUserWorkItem(//异步加载后台数据

o =>

{

try

{

List<YmExecutionorderEx> orders = itransactionSearchBLLService.selectByPage(getCondition(), 1, this.pageSize);

int count = select();

int pageSize = getPageSize();

pageControl.PagingUtil(count, pageSize);

this.refreshGridControl(getResult(orders));

setBestFitColumns();

}

catch (Exception e)

{

LogHelper.Error(e.Message);

}

finally

{

this.setUIStatus(true);//加载数据完毕，界面控件可用，同时隐藏Loading

}

});

}

