DEMO1

 <Grid>

 <Grid.RowDefinitions>

 <RowDefinition Height="Auto" />

 <RowDefinition Height="Auto" />

 </Grid.RowDefinitions>

 <ListView ItemsSource="{Binding Path=rows}">

 <ListView.View>

 <GridView>

 <GridViewColumn Header="名前" CellTemplate="{StaticResource 名前セル}" />

 <GridViewColumn Header="年齢" DisplayMemberBinding="{Binding Path=年齢}" />

 <GridViewColumn Header="性別" DisplayMemberBinding="{Binding Path=性別}" />

 </GridView>

 </ListView.View>

 </ListView>

 <Button Click="clk" Grid.Row="1" Content="クリック!" />

 </Grid>

デバッグしやすくなるように、ボタンを配置しておきます。

ビハインドコードも記述します

ビハインド

 private void clk(object sender, RoutedEventArgs e)

 {

 var x = (Document) this.DataContext;

 x.Click();

 }

ドキュメント

 public class Document : NotifyPropertyChangedBase

 {

 public void Click()

 {

 }

さらさらっと書いて



ハイ拍手！

DEMO2

ソートつけるにはごりごり実装が必要です。

 <ListView ItemsSource="{Binding Path=rows}" x:Name="UserList" GridViewColumnHeader.Click="ヘッダクリック">

コードビハインドからドキュメントクラスへのやり取りがしやすくなるように

 private Document doc{get{ return (Document)this.DataContext;}}

 private void clk(object sender, RoutedEventArgs e)

 {

 this.doc.Click();

 }

べたなコード書きましょう

 private ListSortDirection ソート順 = ListSortDirection.Ascending;

 private string ソートキー = "";

 private void ヘッダクリック(object sender, RoutedEventArgs e)

 {

 var ヘッダ = (GridViewColumnHeader) e.OriginalSource;

 if ( (string)ヘッダ.Column.Header == ソートキー)

 {

 ソート順 = ソート順 == ListSortDirection.Ascending ? ListSortDirection.Descending : ListSortDirection.Ascending;

 }

 else

 {

 ソートキー = (string) ヘッダ.Column.Header;

 ソート順 = ListSortDirection.Ascending;

 }

 var view = CollectionViewSource.GetDefaultView(this.UserList.ItemsSource);

 view.SortDescriptions.Clear();

 view.SortDescriptions.Add(new SortDescription(ソートキー, ソート順));

 view.Refresh();

 }



ソート



成功！！

DEMO3

テンプレートを作ります

 <DataTemplate x:Key="名前セル">

 <Border BorderBrush="LightGray" BorderThickness="1,1,1,1">

 <TextBox Text="{Binding Path=名前}" />

 </Border>

 </DataTemplate>

テンプレートに差し替える

 <GridViewColumn Header="名前" DisplayMemberBinding="{Binding Path=名前}" />

 ↓

 <GridViewColumn Header="名前" CellTemplate="{StaticResource 名前セル}" />



拍手

値が変わっているか確認しましょう

クリック処理にブレークを仕込んで

値を変えて

クリック





DEMO4

Documentクラスを完全に作り替えます。

 public class Document : NotifyPropertyChangedBase

 {

 public Document()

 {

 this.\_rows.Add(new Row {名前 = "なか", 年齢 = 32, 性別 = 性別Enum.男});

 this.\_rows.Add(new Row {名前 = "えムナウ", 年齢 = 21, 性別 = 性別Enum.男});

 this.\_rows.Add(new Row {名前 = "R田中", 年齢 = 18, 性別 = 性別Enum.シラネ});

 }

 public ObservableCollection<Row> rows

 {

 get { return \_rows; }

 set { \_rows = value;

 this.FirePropertyChanged("rows");}

 }

 private ObservableCollection<Row> \_rows = new ObservableCollection<Row>();

 public void Click()

 {

 this.\_rows.Add(new Row { 名前 = "夏椰", 年齢 = 16, 性別 = 性別Enum.女 });

 }

 }





完璧