DEMO1

<Window x:Class="WpfApplication1.Window1"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Window1" Height="300" Width="300" FontSize="24">

<Grid>

<ComboBox IsEditable="true">

<ComboBoxItem>なか</ComboBoxItem>

<ComboBoxItem>えムナウ</ComboBoxItem>

<ComboBoxItem>R田中</ComboBoxItem>

</ComboBox>

</Grid>

</Window>

DEMO2

Pptxでデータの作りを説明

Xamlを変更

ItemSourceがポイント！

<Window x:Class="WpfApplication1.Window1"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:WpfApplication1="clr-namespace:WpfApplication1;assembly="

Title="Window1" Height="300" Width="300" FontSize="24">

<Window.DataContext>

<WpfApplication1:Document />

</Window.DataContext>

<Grid>

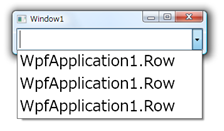
<ComboBox IsEditable="true" ItemsSource="{Binding Path=rows}" />

</Grid>

</Window>

実行

拍手



どうやらうまくバインディングされていて、.ToString()しているような肝心も気がしますね。

それではXAMLを編集してみます。

<Grid>

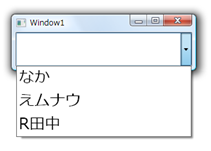
<ComboBox IsEditable="true" ItemsSource="{Binding Path=rows}"

DisplayMemberPath="名前"/>

</Grid>

DisplayMemberPathを追加します。

実行



拍手!!!

それでは選択したデータを取得する方法をやってみましょう。

Xaml側

<ComboBox IsEditable="true" ItemsSource="{Binding Path=rows}"

DisplayMemberPath="名前" SelectedItem="{Binding Path=選択行}" />

Document側

選択行を追加

public Row 選択行

{

get { return \_選択行; }

set { \_選択行 = value;

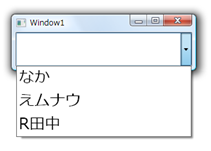
MessageBox.Show(value.年齢.ToString());

}

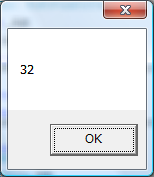
}

private Row \_選択行;

実行



みにくいけど



拍手!!!

他のバインディング方法(Textへのバインディング)

XAML

<ComboBox IsEditable="true" ItemsSource="{Binding Path=rows}"

DisplayMemberPath="名前" SelectedItem="{Binding Path=選択行}" Text="{Binding Path=選択データ}"/>

Document

private string \_選択データ;

public string 選択データ

{

get { return \_選択データ; }

set { \_選択データ = value; this.FirePropertyChanged("選択データ");

MessageBox.Show(value);

}

}

他のバインディング方法(SelectedValueへのバインディング)

とりあえず年齢を連携させたいので、intに変更する

private int \_選択データ;

public int 選択データ

{

get { return \_選択データ; }

set { \_選択データ = value; this.FirePropertyChanged("選択データ");

MessageBox.Show(value.ToString());

}

}

<ComboBox ItemsSource="{Binding Path=rows}"

DisplayMemberPath="名前" SelectedValuePath="年齢" SelectedValue="{Binding Path=選択データ}" />

DEMO3

とりあえずComboBoxを元に戻します

<ComboBox ItemsSource="{Binding Path=rows}" SelectedValue="{Binding Path=選択行}" />

Documentから選択データも消しちゃいます。

Windowのリソースに簡単なテンプレートを作成します。

テンプレートは文字通りよく使うような形を登録しておくようなものです。

<Window.Resources>

<DataTemplate x:Key="DataTemplate1">

<WrapPanel>

<TextBlock Text="{Binding Path=名前}" />

<TextBlock Text="{Binding Path=年齢}" />

</WrapPanel>

</DataTemplate>

</Window.Resources>

こんなものを作成します。

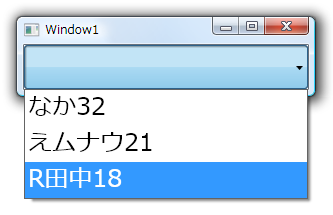
何をしようとしているかわかってきたんじゃないでしょうか。

既存のComboBoxに結びつける方法は簡単

<ComboBox ItemsSource="{Binding Path=rows}"

SelectedValue="{Binding Path=選択行}" ItemTemplate="{DynamicResource DataTemplate1}" />

実行



拍手!!

このままじゃがたがたで不細工なので、ちょっとばかりきれいに細工してみましょう。

<DataTemplate x:Key="DataTemplate2">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="0.6\*"/>

<ColumnDefinition Width="0.1\*"/>

<ColumnDefinition Width="0.3\*"/>

</Grid.ColumnDefinitions>

<TextBlock Text="{Binding Path=名前}" />

<TextBlock Text="{Binding Path=年齢}" Grid.Column="1"/>

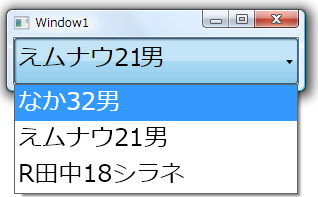
<TextBlock Text="{Binding Path=性別}" Grid.Column="2"/>

</Grid>

</DataTemplate>

<ComboBox ItemsSource="{Binding Path=rows}"

SelectedValue="{Binding Path=選択行}" ItemTemplate="{DynamicResource DataTemplate2}" />



あれー整形したのにうまく行ってないよ~

Pptxに戻る

DEMO4

<DataTemplate x:Key="DataTemplate2">

<Grid Width="{Binding RelativeSource={RelativeSource Mode=FindAncestor, AncestorType={x:Type ComboBox}}, Path=ActualWidth}">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="0.6\*"/>

<ColumnDefinition Width="0.1\*"/>

<ColumnDefinition Width="0.3\*"/>

</Grid.ColumnDefinitions>

<TextBlock Text="{Binding Path=名前}" />

<TextBlock Text="{Binding Path=年齢}" Grid.Column="1"/>

<TextBlock Text="{Binding Path=性別}" Grid.Column="2"/>

</Grid>

</DataTemplate>

実行



拍手

強引にマージンを設定する方法

<ColumnDefinition Width="20"/>