Виктор, проверил вашу работу по последнему уроку. все замечательно. Мне очень нравится.  
По вашему вопросу, нам и тут придется схитрить.  
Я добавлю новое свойство, и новый класс, прокси класс для Департаментов.

Сам прокси класс (на уровне сборки Employees.Communication):

public enum DepartmentProxy

{

[Display(Description = "Основной")]

General = 0,

[Display(Description = "Производственный")]

Industrial = 1,

[Display(Description = "Персонала")]

Personal = 2,

[Display(Description = "Финансов")]

Financial = 3,

}

Измененный класс Employee:

public partial class Employee : ICloneable

{

#region Properties

public DepartmentProxy DepartmentNameProxy

{

get

{

return (DepartmentProxy)DepartmentName;

}

set

{

DepartmentName = (Department)value;

this.RaisePropertyChanged("DepartmentNameProxy");

}

}

#endregion

#region Constructors

public Employee() { }

public Employee(string phone, string firstName, string lastName, string secondName)

{

Phone = phone;

FirstName = firstName;

LastName = lastName;

SecondName = secondName;

}

public Employee(string phone, string firstName, string lastName, string secondName, bool locked, Department departmentName)

{

Phone = phone;

FirstName = firstName;

LastName = lastName;

SecondName = secondName;

Locked = locked;

DepartmentName = departmentName;

}

#endregion

public object Clone()

{

return this.MemberwiseClone();

}

}

Основная идея, свойство DepartmentNameProxy просто является оберткой над DepartmentName и транслирует значение, далее мы работает просто со свойством DepartmentNameProxy и все:

Меняем код в компоненте EmployeesControl :

public ObservableCollection<DepartmentProxy> DepartmentList { get; set; } = new ObservableCollection<DepartmentProxy>();

public EmployeesControl()

{

InitializeComponent();

this.DataContext = this;

DepartmentList.Add(DepartmentProxy.General);

DepartmentList.Add(DepartmentProxy.Industrial);

DepartmentList.Add(DepartmentProxy.Personal);

DepartmentList.Add(DepartmentProxy.Financial);

}

Меняем код в xaml:

<ComboBox x:Name="cbDepardment" ItemsSource="{Binding DepartmentList}"

SelectedValue="{Binding Employee.DepartmentNameProxy}"

SelectedItem="{Binding SelectedValue, Converter={StaticResource DepartmentEnumConverter}, RelativeSource={RelativeSource Self}}"

Margin="119,114,9,0" VerticalAlignment="Top">

<ComboBox.ItemTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=., Converter={StaticResource DepartmentEnumConverter}}"></TextBlock>

</DataTemplate>

</ComboBox.ItemTemplate>

</ComboBox>

И все, теперь у нас все работает, благодаря тому, что мы ввели вспомогательное свойство DepartmentNameProxy.

Виктор, добрый вечер, ошибка заключается в том, что вы создали вспомогательный класс DisplayAttribute.

internal class DisplayAttribute : Attribute  
{  
public string Description { get; set; }  
}

Этот класс необходимо удалить. После удаления подключите сборку System.ComponentModel.DataAnnotations в сборку Employees.Communication.

В этой сборке в неймспейсе System.ComponentModel.DataAnnotations Есть системный атрибут DisplayAttribute, вот его мы и используем:

using System;  
using System.Collections.Generic;  
using System.ComponentModel.DataAnnotations; <-------------------------------------- ВНИМАНИЕ СЮДА  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;

namespace Employees.Communication  
{  
public enum DepartmentProxy  
{  
[Display(Description = "Основной")]  
General = 0,  
[Display(Description = "Производственный")]  
Industrial = 1,  
[Display(Description = "Персонала")]  
Personal = 2,  
[Display(Description = "Финансов")]  
Financial = 3,  
}  
}

Удаляем internal class DisplayAttribute : Attribute, добавляем сборку System.ComponentModel.DataAnnotations в сборку Employees.Communication и все работает.