Привязка к объектам. Интерфейс INotifyPropertyChanged

Последнее обновление: 21.06.2016

Если в качестве цели привязки обязательно должен выступать объект класса BindableObject, то источником привязке может быть любой объект, в том числе самого стандартного класса. Однако источником является простой класс, не представляющий BindableObject, то мы можем столкнуться с проблемой обновления данных при привязке. Например, пусть у нас в главном проекте есть класс Phone:

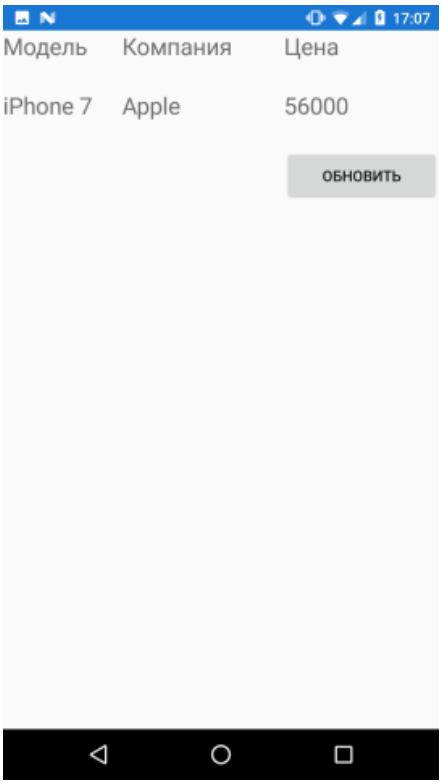
```
public class Phone
{
    public string Title { get; set; }
    public string Company { get; set; }
    public int Price { get; set; }
}
```

Тогда осуществим привязку к объекту класса Phone:

```
using System;
using Xamarin.Forms;

namespace HelloApp
{
    public partial class MainPage : ContentPage
    {
        Phone phone;
        public MainPage()
        {
            phone = new Phone { Title = "iPhone 7", Company = "Apple", Price = 56000 };

            Grid grid = new Grid
            {
                RowDefinitions =
                {
                    new RowDefinition { Height = 50 },
                    new RowDefinition { Height = 50 },
                    new RowDefinition { Height = 50 }
                },
                ColumnDefinitions =
                {
                    new ColumnDefinition { Width = new GridLength(0.8, GridUnitType.Star) },
                    new ColumnDefinition { Width = new GridLength(1.1, GridUnitType.Star) },
                    new ColumnDefinition { Width = new GridLength(1.1, GridUnitType.Star) }
                }
            };

            Label titleHeaderLabel = new Label { Text = "Модель" };
            Label companyHeaderLabel = new Label { Text = "Компания" };
            Label priceHeaderLabel = new Label { Text = "Цена" };

            Label titleValueLabel = new Label();
            Binding titleBinding = new Binding { Source = phone, Path = "Title" };
            titleValueLabel.SetBinding(Label.TextProperty, titleBinding);

            Label companyValueLabel = new Label();
            Binding companyBinding = new Binding { Source = phone, Path = "Company" };
            companyValueLabel.SetBinding(Label.TextProperty, companyBinding);

            Label priceValueLabel = new Label();
            Binding priceBinding = new Binding { Source = phone, Path = "Price" };
```

```
            priceValueLabel.SetBinding(Label.TextProperty, priceBinding);

            Button updateButton = new Button { Text = "Обновить" };
            updateButton.Clicked += UpdateButton_Clicked;

            grid.Children.Add(titleHeaderLabel, 0, 0);
            grid.Children.Add(companyHeaderLabel, 1, 0);
            grid.Children.Add(priceHeaderLabel, 2, 0);

            grid.Children.Add(titleValueLabel, 0, 1);
            grid.Children.Add(companyValueLabel, 1, 1);
            grid.Children.Add(priceValueLabel, 2, 1);

            grid.Children.Add(updateButton, 2, 2);

            Content = grid;
        }

        private void UpdateButton_Clicked(object sender, EventArgs e)
        {
            phone.Price += 4000;
        }
    }
}
```

Здесь на странице три элемента Label привязаны к разным свойствам объекта Phone. В итоге при запуске приложения все три метки отобразят значения свойств Phone:



Однако если мы нажмем на кнопку, то мы никаких изменений не увидим. Хотя обработчик кнопки изменит значение свойства Price объекта Phone. Однако соответствующий элемент Label не изменит свой тест, так как он просто не будет знать, что привязанное свойство Price изменилось.

И чтобы уведомить цель привязки об изменении значений источник привязки должен реализовать интерфейс **INotifyPropertyChanged**. Для этого изменим класс Phone следующим образом:

```
using System.ComponentModel;

namespace HelloApp
{
    public class Phone : INotifyPropertyChanged
    {
        private string title;
        private string company;
        private int price;

        public string Title
        {
            get { return title; }
```

```
            set
            {
                if (title != value)
                {
                    title = value;
                    OnPropertyChanged("Title");
                }
            }
        }
        public string Company
        {
            get { return company; }
            set
            {
                if (company != value)
                {
                    company = value;
                    OnPropertyChanged("Company");
                }
            }
        }
        public int Price
        {
            get { return price; }
            set
            {
                if (price != value)
                {
                    price = value;
                    OnPropertyChanged("Price");
                }
            }
        }

        public event PropertyChangedEventHandler PropertyChanged;
        public void OnPropertyChanged(string prop = "")
        {
            if (PropertyChanged != null)
                PropertyChanged(this, new PropertyChangedEventArgs(prop));
        }
    }
}
```

При этом в коде самой страницы MainPage ничего менять не надо. И если теперь мы нажмем на кнопку, то элемент Label мгновенно отреагирует на изменение свойства Price объекта Phone.

Аналогичный пример в XAML:

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:HelloApp;assembly=HelloApp"
             x:Class="HelloApp.MainPage">
  <ContentPage.Resources>
    <ResourceDictionary>
      <local:Phone x:Key="phone" Title="iPhone 7" Company ="Apple" Price="56000" />
    </ResourceDictionary>
  </ContentPage.Resources>
  <Grid BindingContext="{StaticResource phone}">
    <Grid.ColumnDefinitions>
      <ColumnDefinition Width="0.8*" />
      <ColumnDefinition Width="1.1*" />
      <ColumnDefinition Width="1.1*" />
    </Grid.ColumnDefinitions>
    <Grid.RowDefinitions>
      <RowDefinition Height="50" />
      <RowDefinition Height="50" />
      <RowDefinition Height="50" />
    </Grid.RowDefinitions>
    <Label Grid.Column="0" Grid.Row="0" Text="Модель" />
    <Label Grid.Column="1" Grid.Row="0" Text="Компания"  />
    <Label Grid.Column="2" Grid.Row="0" Text="Цена"   />

    <Label Grid.Column="0" Grid.Row="1" Text="{Binding Path=Title}" />
    <Label Grid.Column="1" Grid.Row="1" Text="{Binding Path=Company}"  />
    <Label Grid.Column="2" Grid.Row="1" Text="{Binding Path=Price}"   />

    <Button Text="Обновить" Grid.Column="2" Grid.Row="2" Clicked="UpdateButton_Clicked" />
  </Grid>
</ContentPage>
```

Здесь объект Phone определяется как статический ресурс. Затем мы можем задать его как контекст для всего элемента Grid, а в отдельных элементах Label прописать привязку к свойствам этого объекта.

А в классе связанного кода пропишем обработчик кнопки, который будет менять значение свойства Price:

```
using System;
using Xamarin.Forms;

namespace HelloApp
{
    public partial class MainPage : ContentPage
    {
        public MainPage()
        {
            InitializeComponent();
        }

        private void UpdateButton_Clicked(object sender, EventArgs e)
        {
            var phone = this.Resources["phone"] as Phone;
            phone.Price += 4000;
        }
    }
}
```


Назад Содержание Вперед

AZinX • 5 месяцев назад

Помогите разобраться.
Есть страница, есть ViewModel этой страницы.

На странице есть Label и кнопка. Суть в том, чтобы по нажатию на кнопку метка исчезала и появлялась

На странице есть Label и кнопка. Суть в том, чтобы по нажатию на кнопку метка исчезала и появлялась.
Я реализовал это через паттерн MVVM и интерфейс комманд, но почему-то не работает. Но если реализовывать через событие Clicked у кнопки, то все работает. В чем может быть проблема? Прилагаю листинги.
---
xaml страницы TestPage.xaml.

```
<contentpage xmlns="http://xamarin.com/schemas/..." xmlns:x="http://schemas.microsoft.co..."
x:class="TestApp.Views.TestPage">

<stacklayout>
<label text="Информация" fontsize="Medium" isvisible="{Binding IsVisibleInfo}"/>
<button text="i" command="{Binding InfoCommand}"/>
</stacklayout>

</contentpage>
```

показать больше

^ | v • Ответить • Поделиться ›

AZinX ➚ AZinX • 5 месяцев назад
Все, разобрался. Ой смешно просто...
Строка которую мы передаем в OnPropertyChanged - это название свойства.
Я просто думал, что туда можно любую строку передать и система поймет, что произошло изменение.
^ | v • Ответить • Поделиться ›

Metanit Модератор ➚ AZinX • 5 месяцев назад
а вы отладкой проходились по коду?
^ | v • Ответить • Поделиться ›

AZinX ➚ Metanit • 5 месяцев назад
Хм, оказывается можно отлаживать :)
Так вот, при загрузке страницы поле infovisibility устанавливается в false; Но потом, при нажатии на кнопку, брейкпойнт не срабатывает.

Очень странная штука. Если поставить брейкпойнт на свойство, то при нажатии на кнопку вызывается команда, которая устанавливает свойство, которое в свою очередь устанавливает поле infovisibility.
Вызывается и метод OnPropertyChanged. Все как надо. Но Label все равно не появляется.

Что-то совсем неясно.
^ | v • Ответить • Поделиться ›

AZinX ➚ Metanit • 5 месяцев назад
Я тестирую на реальном устройстве, а там же вроде нельзя отладкой походить.
^ | v • Ответить • Поделиться ›

Андрей Петров • 9 месяцев назад
У Вас ошибка(описка) в классе Phone с уже реализованным интерфейсом "INotifyPropertyChanged", вместо свойства "Name" должно быть "Title"
^ | v • Ответить • Поделиться ›

Metanit Модератор ➚ Андрей Петров • 9 месяцев назад
ага, поправил
^ | v • Ответить • Поделиться ›