Министерство образования Московской области

Государственное образовательное учреждение высшего образования Московской области

«Государственный гуманитарно-технологический университет»

**Ликино-Дулевский политехнический колледж – филиал ГГТУ**

**ПРАКТИЧЕСКАЯ РАБОТА**

**Binding/Задачи.**

по дисциплине

«МДК.01.03 Разработка мобильных приложений»

**Выполнил:**

Рылеев Александр Юрьевич

студент группы

ИСП.20А

09.02.07 Информационные системы и программирование

очной формы обучения

Проверила:

Гжегожевский Сергей Владимирович

Оценка

\_\_\_\_\_\_\_\_\_/\_\_ \_\_\_\_\_\_\_\_\_\_/

1. Постановка задачи

**Задание 1. Простая привязка.**

Создайте следующий интерфейс:

|  |  |
| --- | --- |
|  | Сверху разместите 3 объекта Entry, затем кнопку, затем 3 Label.  Объект Entry выступают как источники (source).  Объекты Label соответственно, как цель (target)  Для данной задачи не создавайте дополнительный класс.  При нажатии на кнопку данные из Entry должны меняться друг с другом (1->2, 2->3, 3->1)  Решите задачу двумя способами:   1. Через привязку из программного кода. 2. Через привязку из xaml   Код 1 и 2 способов вставьте в отчет к заданию.  Так же приложите скрин работающего приложения.  Ссылки на теорию и примеры:  <https://metanit.com/sharp/maui/6.1.php>  <https://metanit.com/sharp/maui/6.2.php> |

**Задание 2. Простая привязка.**

|  |  |
| --- | --- |
|  | Сверху разместите 3 объекта Entry, затем кнопку, затем ListView.  Создайте класс Car с 3 атрибутами: Company, Model, Price.  При нажатии на кнопку данные добавляются и отображаются в списке под кнопкой.  Сделайте данную задачу по примеру:  <https://metanit.com/sharp/maui/8.6.php>  Используйте любой метод привязки: через код или через xaml.    Программный код вставьте в отчет к заданию.  Так же приложите скрин работающего приложения. |

**Задание 3. Марки Авто.**

|  |  |
| --- | --- |
|  | Необходимо вывести список об автомобилях с логотипом компании.  Сделайте данную задачу по примеру:  <https://metanit.com/sharp/maui/8.4.php>  Используйте любой метод привязки: через код или через xaml.    Программный код вставьте в отчет к заданию.  Так же приложите скрин работающего приложения. |

1. Реализация: Задание №1.1

1) Напишем следующий код в «Binding1.xaml.cs»:

namespace BindingApp;

public partial class Binding1 : ContentPage

{

// Создание полей ввода и меток

Entry entry1 = new Entry()

{

Margin = 5,

WidthRequest = 300,

HeightRequest = 20

}, entry2 = new Entry()

{

Margin = 5,

WidthRequest = 300,

HeightRequest = 20

}, entry3 = new Entry()

{

Margin = 5,

WidthRequest = 300,

HeightRequest = 20

};

Label label1 = new Label()

{

Margin = 5,

WidthRequest = 300,

HeightRequest = 20,

Background = new SolidColorBrush(Colors.White),

TextColor = Colors.Black

}, label2 = new Label()

{

Margin = 5,

WidthRequest = 300,

HeightRequest = 20,

Background = new SolidColorBrush(Colors.White),

TextColor = Colors.Black

}, label3 = new Label()

{

Margin = 5,

WidthRequest = 300,

HeightRequest = 20,

Background = new SolidColorBrush(Colors.White),

TextColor = Colors.Black

};

public Binding1()

{

// Создание кнопки и события для неё

Button button = new Button { Text = "Поменять", WidthRequest = 150, HeightRequest = 45, Margin = 5 };

button.Clicked += Button\_Clicked;

// Устанавливаем привязку

// источник привязки - entry, цель привязки - label

label1.BindingContext = entry1;

label2.BindingContext = entry2;

label3.BindingContext = entry3;

// Связываем свойства источника и цели

label1.SetBinding(Label.TextProperty, "Text");

label2.SetBinding(Label.TextProperty, "Text");

label3.SetBinding(Label.TextProperty, "Text");

// определяем объект привязки: Source - источник, Path - его свойство

Binding binding1 = new Binding { Source = entry1, Path = "Text" };

Binding binding2 = new Binding { Source = entry2, Path = "Text" };

Binding binding3 = new Binding { Source = entry3, Path = "Text" };

// установка привязки для свойства TextProperty

label1.SetBinding(Label.TextProperty, binding1);

label2.SetBinding(Label.TextProperty, binding2);

label3.SetBinding(Label.TextProperty, binding3);

StackLayout stackLayout = new StackLayout()

{

Children = { entry1, entry2, entry3, button, label1, label2, label3 },

Padding = 20

};

Content = stackLayout;

}

// Событие для кнопки

private void Button\_Clicked(object sender, EventArgs e)

{

string s1 = entry1.Text;

string s2 = entry2.Text;

string s3 = entry3.Text;

entry3.Text = s2;

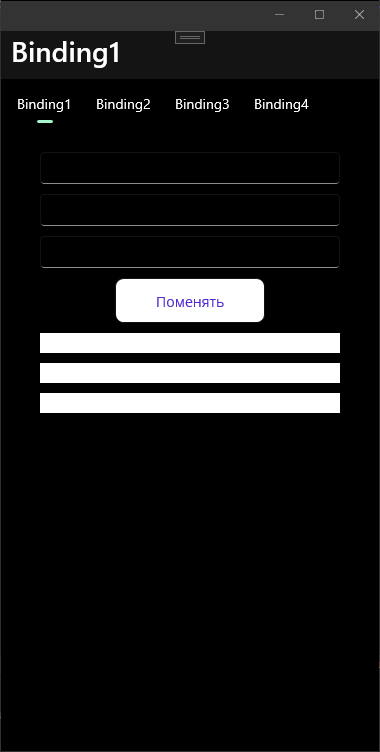
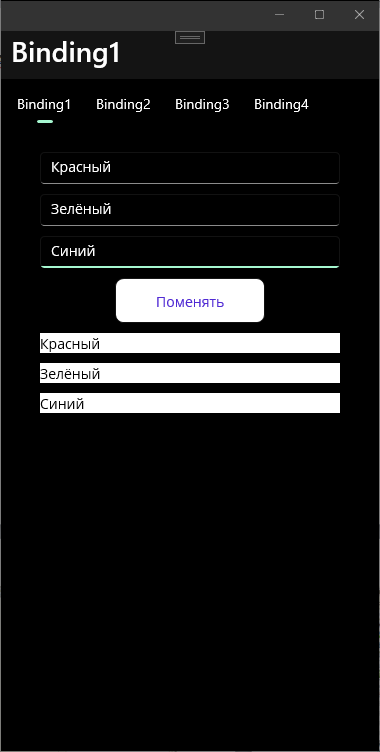
entry1.Text = s3;

entry2.Text = s1;

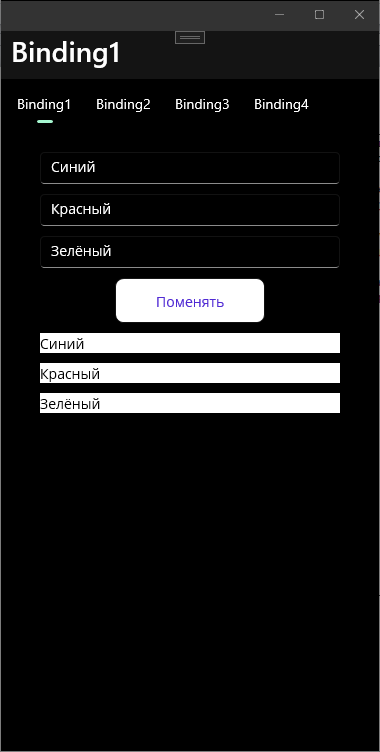
}

}

2) Интерфейс готов:

3) Проверим функционал, изменение полей при нажатии на кнопку «Поменять»:

1. Реализация: Задание №1.2

1) Напишем следующий код в «Binding2.xaml»:

<?xml version="1.0" encoding="utf-8" ?>

<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"

x:Class="BindingApp.Binding2"

Title="Binding2">

<StackLayout Padding="20">

<Entry WidthRequest="300" HeightRequest="20" Margin="5" x:Name="entryBox1" />

<Entry WidthRequest="300" HeightRequest="20" Margin="5" x:Name="entryBox2" />

<Entry WidthRequest="300" HeightRequest="20" Margin="5" x:Name="entryBox3" />

<Button WidthRequest="150" HeightRequest="45" Margin="5" Text="Поменять" Clicked="Button\_Clicked"/>

<Label WidthRequest="300" HeightRequest="20" Margin="5" Background="White" TextColor="Black" BindingContext="{x:Reference entryBox1}" Text="{Binding Text}" />

<Label WidthRequest="300" HeightRequest="20" Margin="5" Background="White" TextColor="Black" BindingContext="{x:Reference entryBox2}" Text="{Binding Text}" />

<Label WidthRequest="300" HeightRequest="20" Margin="5" Background="White" TextColor="Black" BindingContext="{x:Reference entryBox3}" Text="{Binding Text}" />

</StackLayout>

</ContentPage>

2) Напишем следующий код в «Binding2.xaml.cs»:

namespace BindingApp;

public partial class Binding2 : ContentPage

{

public Binding2()

{

InitializeComponent();

}

// Событие для кнопки

private void Button\_Clicked(object sender, EventArgs e)

{

string s1 = entryBox1.Text;

string s2 = entryBox2.Text;

string s3 = entryBox3.Text;

entryBox3.Text = s2;

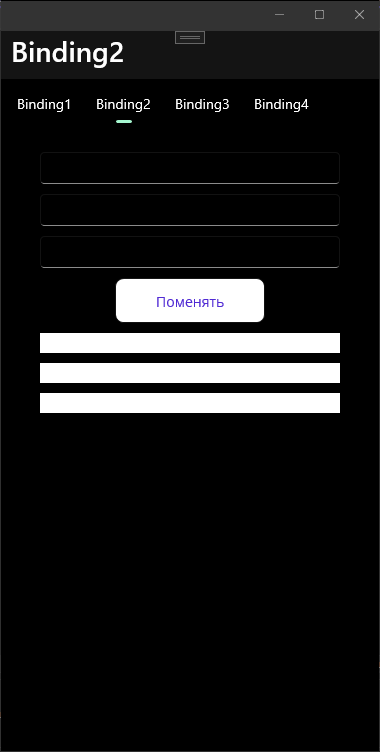
entryBox1.Text = s3;

entryBox2.Text = s1;

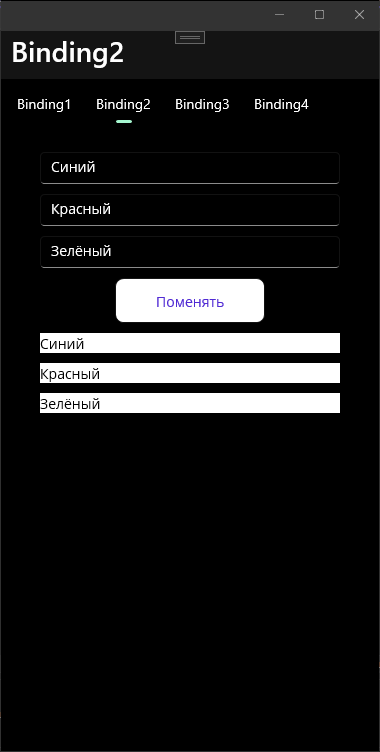
}

}

3) Интерфейс готов:

4) Проверим функционал, изменение полей при нажатии на кнопку «Поменять»:

1. Реализация: Задание №2

1) Напишем следующий код в «Binding3.xaml.cs»:

namespace BindingApp;

public partial class Binding3 : ContentPage

{

// Создание класса Car

public class Car

{

public string Company { get; set; } = "";

public string Model { get; set; } = "";

public int Price { get; set; }

}

// Список машин

public List<Car> Cars { get; set; }

// Список, отображаемый на форме

ListView listView = new ListView();

// Создание полей ввода

Entry companyEntry, modelEntry, priceEntry;

public Binding3()

{

// определяем данные

Cars = new List<Car>

{

new Car {Company = "Lada", Model="Granta", Price = 75000},

new Car {Company = "Audi", Model="A5", Price = 1750000},

new Car {Company = "BMW", Model="X6", Price = 2350000}

};

// определяем источник данных

listView.ItemsSource = Cars;

// определяем шаблон данных

listView.ItemTemplate = new DataTemplate(() =>

{

// привязка к свойству Company

Label companyLabel = new Label { Margin = new Thickness(15, 0), WidthRequest = 75, HorizontalOptions = LayoutOptions.Center };

companyLabel.SetBinding(Label.TextProperty, "Company");

// привязка к свойству Model

Label modelLabel = new Label { Margin = new Thickness(15, 0), WidthRequest = 75, HorizontalOptions = LayoutOptions.Center };

modelLabel.SetBinding(Label.TextProperty, "Model");

// привязка к свойству Price

Label priceLabel = new Label { Margin = new Thickness(15, 0), WidthRequest = 75, HorizontalOptions = LayoutOptions.Center };

priceLabel.SetBinding(Label.TextProperty, "Price");

// создаем объект ViewCell.

return new ViewCell

{

View = new StackLayout

{

WidthRequest = 300,

HeightRequest = 35,

Padding = new Thickness(0, 5),

Orientation = StackOrientation.Horizontal,

Children = { companyLabel, modelLabel, priceLabel }

}

};

});

// поля для добавления нового объекта User

companyEntry = new Entry { Placeholder = "Enter company", Margin = 5, WidthRequest = 300, HeightRequest = 20 };

modelEntry = new Entry { Placeholder = "Enter model", Margin = 5, WidthRequest = 300, HeightRequest = 20 };

priceEntry = new Entry { Placeholder = "Enter price", Margin = 5, WidthRequest = 300, HeightRequest = 20 };

Button saveButton = new Button { Text = "Save", WidthRequest = 150, HeightRequest = 45, Margin = 5 };

saveButton.Clicked += SaveButton\_Clicked;

StackLayout form = new StackLayout

{

Orientation = StackOrientation.Vertical,

Children = { companyEntry, modelEntry, priceEntry, saveButton }

};

Content = new StackLayout { Orientation = StackOrientation.Vertical, Children = { form, listView }, Padding = 7 };

}

// Событие для кнопки

private void SaveButton\_Clicked(object sender, EventArgs e)

{

int.TryParse(priceEntry.Text, out var price);

Cars.Add(new Car { Company = companyEntry.Text, Model = modelEntry.Text, Price = price });

listView.ItemsSource = null;

companyEntry.Text = modelEntry.Text = priceEntry.Text = "";

listView.ItemsSource = Cars;

}

}

2) Интерфейс готов:

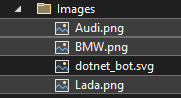
 

3) Проверим функционал, добавление данных и отображение в списке при нажатии на кнопку «Добавить»:

1. Реализация: Задание №3

1) Добавляем скачанные картинки в папку «Images»:





2) Напишем следующий код в «Binding4.xaml.cs»:

namespace BindingApp;

public partial class Binding4 : ContentPage

{

// Создание класса Car

public class Car

{

public string Company { get; set; } = "";

public string Model { get; set; } = "";

public int Price { get; set; }

public string ModelAndPrice { get { return Model + " от " + Price; } }

public string ImagePath { get; set; } = "";

}

public Binding4()

{

ListView listView = new ListView();

// определяем источник данных

listView.ItemsSource = new List<Car>

{

new Car {Company = "Lada", ImagePath="Lada.png", Model="Granta", Price = 75000},

new Car {Company = "Audi", ImagePath="Audi.png", Model="A5", Price = 1750000},

new Car {Company = "BMW", ImagePath="BMW.png", Model="X6", Price = 2350000}

};

// определяем шаблон данных

listView.ItemTemplate = new DataTemplate(() =>

{

ImageCell imageCell = new ImageCell

{

TextColor = Colors.Black,

DetailColor = Colors.Black

};

imageCell.SetBinding(ImageCell.TextProperty, "Company");

imageCell.SetBinding(ImageCell.DetailProperty, "ModelAndPrice");

imageCell.SetBinding(ImageCell.ImageSourceProperty, "ImagePath");

return imageCell;

});

Label header = new Label { FontSize = 20, Text = "Автомобили", TextColor = Colors.Black, FontAttributes = FontAttributes.Bold, Margin = new Thickness(0, 15), HorizontalOptions = LayoutOptions.Center };

Content = new StackLayout { Background=Colors.Wheat, Children = { header, listView }, Padding = 7 };

}

}

3) Интерфейс готов:

