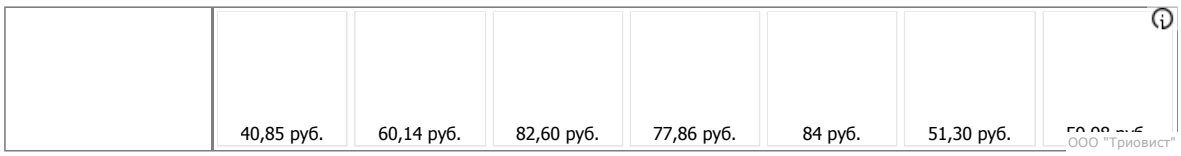
Создание класса ячейки для ListView

Последнее обновление: 29.03.2017



Если визуальное представление, используемое для вывода данных в ListView, повторяется и применяется в отдельных частях приложения, то мы можем вынести это представление в отдельный класс, который унаследован от ViewCell.

Например, нам бы хотелось, чтобы каждая ячейка была наподобие ImageCell, только с возможностью устанавливать высоту и ширину изображения.

Для этого добавим в проект новый класс CustomCell, который будет представлять отдельную ячейку:

```
public class CustomCell : ViewCell
{
    Label titleLabel, detailLabel;
    Image image;

    public CustomCell()
    {
        titleLabel = new Label { FontSize = 18 };
        detailLabel = new Label();
        image = new Image();

        StackLayout cell = new StackLayout();
        cell.Orientation = StackOrientation.Horizontal;

        StackLayout titleDetailLayout = new StackLayout();
        titleDetailLayout.Children.Add(titleLabel);
        titleDetailLayout.Children.Add(detailLabel);

        cell.Children.Add(image);
        cell.Children.Add(titleDetailLayout);
        View = cell;
    }

    public static readonly BindableProperty TitleProperty =
        BindableProperty.Create("Title", typeof(string), typeof(CustomCell), "");

    public static readonly BindableProperty ImagePathProperty =
        BindableProperty.Create("ImagePath", typeof(ImageSource), typeof(CustomCell), null);

    public static readonly BindableProperty ImageWidthProperty =
        BindableProperty.Create("ImageWidth", typeof(int), typeof(CustomCell), 100);

    public static readonly BindableProperty ImageHeightProperty =
        BindableProperty.Create("ImageHeight", typeof(int), typeof(CustomCell), 100);

    public static readonly BindableProperty DetailProperty =
        BindableProperty.Create("Detail", typeof(string), typeof(CustomCell), "");

    public string Title
    {
        get { return (string)GetValue(TitleProperty); }
        set { SetValue(TitleProperty, value); }
    }

    public int ImageWidth
    {
        get { return (int)GetValue(ImageWidthProperty); }
        set { SetValue(ImageWidthProperty, value); }
```

```
    }
    public int ImageHeight
    {
        get { return (int)GetValue(ImageHeightProperty); }
        set { SetValue(ImageHeightProperty, value); }
    }

    public ImageSource ImagePath
    {
        get { return (ImageSource)GetValue(ImagePathProperty); }
        set { SetValue(ImagePathProperty, value); }
    }

    public string Detail
    {
        get { return (string)GetValue(DetailProperty); }
        set { SetValue(DetailProperty, value); }
    }

    protected override void OnBindingContextChanged()
    {
        base.OnBindingContextChanged();

        if (BindingContext != null)
        {
            titleLabel.Text = Title;
            detailLabel.Text = Detail;
            image.Source = ImagePath;
            image.WidthRequest = ImageWidth;
            image.HeightRequest = ImageHeight;
        }
    }
}
```

Ячейка будет состоять из двух элементов Label, расположенных вертикально, и элемента Image.

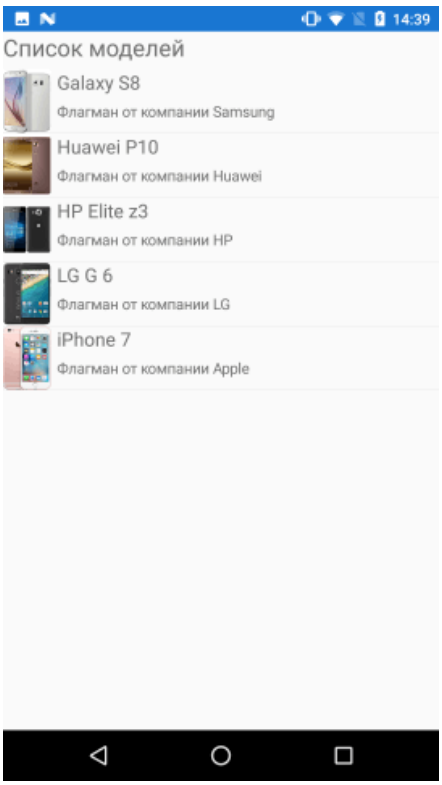
Для получения данных извне и привязки данных к элементам Label и Image определены свойства BindableProperty.

Для обработки изменения контекста привязки переопределен метод `OnBindingContextChanged`.

Определим в коде C# страницу, которая будет выводить список:

```
using System.Collections.Generic;
using Xamarin.Forms;

namespace HelloApp
{
    public partial class MainPage : ContentPage
    {
        public List<Phone> Phones { get; set; }

        public MainPage()
        {
            Phones = new List<Phone>
            {
                new Phone {Title="Galaxy S8", Company="Samsung", Price=48000, ImagePath="galaxys6.jpg" },
                new Phone {Title="Huawei P10", Company="Huawei", Price=35000, ImagePath="mate8.jpg" },
                new Phone {Title="HP Elite z3", Company="HP", Price=42000, ImagePath="lumia950.jpg" },
                new Phone {Title="LG G 6", Company="LG", Price=42000, ImagePath="nexus5x.jpg" },
                new Phone {Title="iPhone 7", Company="Apple", Price=52000, ImagePath="iphone6s.jpg" }
            };
            Label header = new Label
            {
                Text = "Список моделей",
                FontSize = Device.GetNamedSize(NamedSize.Large, typeof(Label))
            };

            ListView listView = new ListView
            {
                HasUnevenRows = true,
                ItemsSource = Phones,

                ItemTemplate = new DataTemplate(() =>
                {
                    CustomCell customCell = new CustomCell { ImageHeight=60, ImageWidth=45 };
                    customCell.SetBinding(CustomCell.TitleProperty, "Title");
                    Binding companyBinding = new Binding { Path = "Company", StringFormat = "Флагман от компании {0}"
};
                    customCell.SetBinding(CustomCell.DetailProperty, companyBinding);
                    customCell.SetBinding(CustomCell.ImagePathProperty, "ImagePath");
```

```
                    return customCell;
                })
            };
            this.Content = new StackLayout { Children = { header, listView } };
        }
    }
    public class Phone
    {
        public string Title { get; set; }
        public string ImagePath { get; set; }
        public string Company { get; set; }
        public int Price { get; set; }
    }
}
```



Использование CustomCell 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">
    <StackLayout>
        <Label Text="{Binding Source={x:Reference Name=phonesList}, Path=SelectedItem.Title}"
            FontSize="Large" />
        <ListView x:Name="phonesList"
              HasUnevenRows="True"
              ItemsSource="{Binding Phones}">
            <ListView.ItemTemplate>
                <DataTemplate>
                    <local:CustomCell
                        ImagePath="{Binding ImagePath}"
                        ImageWidth="45"
                        ImageHeight="60"
                        Title="{Binding Title}"
                        Detail="{Binding Company, StringFormat='Флагман компании {0}'}"/>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
    </StackLayout>
</ContentPage>
```

И связанный код C#:

```
public partial class MainPage : ContentPage
{
    public List<Phone> Phones { get; set; }

    public MainPage()
```

```
    {
        InitializeComponent();
        Phones = new List<Phone>
        {
            new Phone {Title="Galaxy S8", Company="Samsung", Price=48000, ImagePath="galaxys6.jpg" },
            new Phone {Title="Huawei P10", Company="Huawei", Price=35000, ImagePath="mate8.jpg" },
            new Phone {Title="HP Elite z3", Company="HP", Price=42000, ImagePath="lumia950.jpg" },
            new Phone {Title="LG G 6", Company="LG", Price=42000, ImagePath="nexus5x.jpg" },
            new Phone {Title="iPhone 7", Company="Apple", Price=52000, ImagePath="iphone6s.jpg" }
        };
        this.BindingContext = this;
    }
}
```


Назад Содержание Вперед