TextCell

Последнее обновление: 29.03.2017

В прошлой теме для настройки вывода списка в ListView использовался объект ViewCell. Этот объект представляет ячейку, которую мы можем настроить по своему усмотрению, опередилить в ней различные элементы. Однако кроме ViewCell в ListView для отображения данных можно применять еще несколько типов ячеек:

- **TextCell**: выводит заголовок и некоторое детальное описание
- **ImageCell**: выводит рядом с заголовком и детальным описанием изображение

Для простых случаев, когда для каждого объекта в списке необходимо вывести заголовок и некоторую аннотацию к нему, можно использовать класс TextCell. Его основные свойства, которые могут нам пригодиться:

- **Text**: основной текст, выводится большим шрифтом
- **Detail**: детальное описание, выводится меньшим шрифтом
- **TextColor**: цвет текста
- **DetailColor**: цвет детального описания

Например, определим в файле кода класс Phone и список объектов:

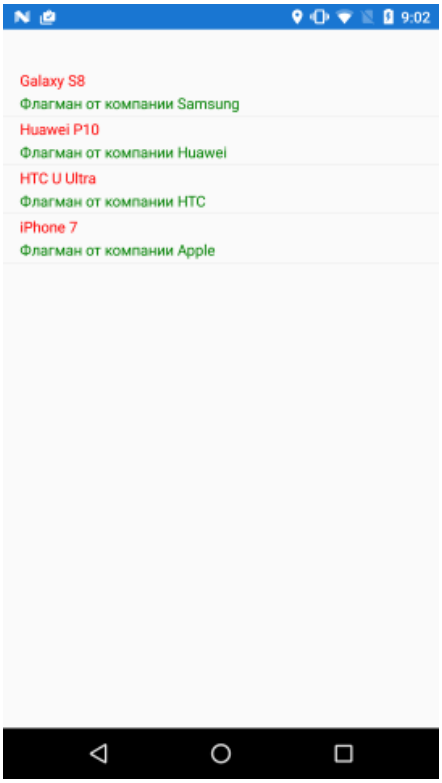
```
using System.Collections.Generic;
using Xamarin.Forms;

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

    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 },
                new Phone {Title="Huawei P10", Company="Huawei", Price=35000 },
                new Phone {Title="HTC U Ultra", Company="HTC", Price=42000 },
                new Phone {Title="iPhone 7", Company="Apple", Price=52000 }
            };
            this.BindingContext = this;
        }
        public async void OnItemTapped(object sender, ItemTappedEventArgs e)
        {
            Phone selectedPhone = e.Item as Phone;
            if (selectedPhone != null)
                await DisplayAlert("Выбранная модель", $"{selectedPhone.Company} - {selectedPhone.Title}", "OK");
        }
```

```
    }
}
```

Настроим привязку и вывод объектов в файле 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}"
              ItemTapped="OnItemTapped">
            <ListView.ItemTemplate>
                <DataTemplate>
                    <TextCell
                        Text="{Binding Title}"
                        Detail="{Binding Company, StringFormat='Флагман от компании {0}'}"
                        TextColor="Red"
                        DetailColor="Green"
                        />
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
    </StackLayout>
</ContentPage>
```

В отличие от прошлой темы в плане привязки ничего не изменяется, только заменяется ViewCell на TextCell:



Но естественно возможностей по кастомизации меньше, чем в случае с ViewCell.

Аналогичный пример полность в коде C#:

```
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 },
                new Phone {Title="Huawei P10", Company="Huawei", Price=35000 },
                new Phone {Title="HTC U Ultra", Company="HTC", Price=42000 },
                new Phone {Title="iPhone 7", Company="Apple", Price=52000 }
            };
            Label header = new Label
            {
                Text = "Список моделей",
                FontSize = Device.GetNamedSize(NamedSize.Large, typeof(Label))
            };

            ListView listView = new ListView
            {
                HasUnevenRows = true,
                // Определяем источник данных
                ItemsSource = Phones,

                // Определяем формат отображения данных
                ItemTemplate = new DataTemplate(() =>
                {
                    // создаем объект TextCell
                    TextCell textCell = new TextCell { TextColor = Color.Red, DetailColor = Color.Green };
                    textCell.SetBinding(TextCell.TextProperty, "Title");
                    Binding companyBinding = new Binding { Path = "Company", StringFormat="Флагман от компании {0}"};
                    textCell.SetBinding(TextCell.DetailProperty, companyBinding);
                    return textCell;
                })
            };
            listView.ItemTapped += OnItemTapped;
            this.Content = new StackLayout { Children = { header, listView } };
        }
        public async void OnItemTapped(object sender, ItemTappedEventArgs e)
        {
            Phone selectedPhone = e.Item as Phone;
            if (selectedPhone != null)
                await DisplayAlert("Выбранная модель", $"{selectedPhone.Company} - {selectedPhone.Title}", "OK");
        }
    }
    public class Phone
    {
        public string Title { get; set; }
        public string Company { get; set; }
        public int Price { get; set; }
    }
}
```


Назад **Содержание** **Вперед**