



TextCell

Последнее обновление: 29.03.2017



Помогите им найти вашу
компанию в Google!

НАЧНИТЕ СЕЙЧАС

Компенсируем
до \$60

Google AdWords

В прошлой теме для настройки вывода списка в ListView использовался объект ViewCell. Этот объект представляет ячейку, которую мы можем настроить по своему усмотрению, опеределить 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;
        }

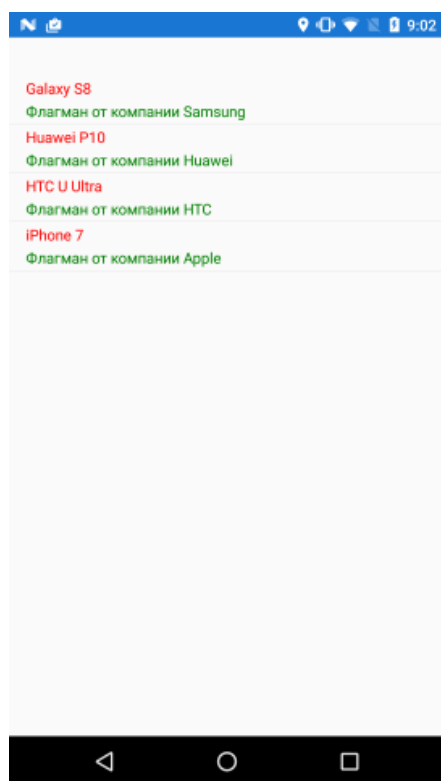
        public async void OnItemTapped(object sender, ItemTappedEventArgs e)
        {
            Phone selectedPhone = e.Item as Phone;
            if (selectedPhone != null)
                await DisplayAlert("Выбранная модель", $"{selectedPhone.Company} - {selectedPhone.Title}", "OK");
        }
    }
}
```

```
}
}
```

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 полностью 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; }
}
}

```



Зимние шины в Минске
255/55R18

avd.by



Яндекс.Директ

[Назад](#) [Содержание](#) [Вперед](#)





2 Комментариев metanit.com

1 Войти ▾

Рекомендовать
 Поделиться

Лучшее в начале ▾



Присоединиться к обсуждению...

ВОЙТИ С ПОМОЩЬЮ

или ЧЕРЕЗ DISQUS ?

Имя

**Андрей Лысенко** • месяц назад

Хотелось бы, чтобы Вы уточнили в чем больше возможностей по кастомизации у ViewCell нежели чем у TextCell

^ | ▾ • Ответить • Поделиться ›

**Андрей Лысенко** ➔ Андрей Лысенко • месяц назад

Я извиняюсь за глупый вопрос, дошло как до жирафа, viewcell принимает объект view, в который все что угодно можно залепить, контейнер с вложениями

^ | ▾ • Ответить • Поделиться ›

ТАКЖЕ НА METANIT.COM

Паттерны в C# и .NET | Fluent Builder

10 комментариев • 3 месяца назад

**LamaK** — При том, что, если забыть какой-то параметр, то что-то может обвалиться. По-хорошему, в метод Build() можно добавить проверку всех необходимых ...**Go | Visual Studio Code**

1 комментарий • месяц назад

**zeldenis** — В первом предложении опечатка "подцветку".**C# и .NET | Раннее и позднее связывание**

3 комментариев • 2 месяца назад

**dev loop** — спасибо, ответили и на мой вопрос тоже)**Go | Сервер. Обработка подключений**

2 комментариев • 7 дней назад

**Metanit** — Да

Подписаться

 Добавить Disqus на свой сайт
 Добавить Disqus
 Добавить
 Конфиденциальность