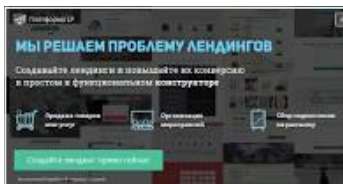




## Контекстное меню

Последнее обновление: 29.08.2016

**Удобный  
конструктор сайтов**

Запусти свой сайт баз  
навыков программирования  
за час. Бесплатный период  
14 дней.



Ключевым элементом для работы с наборами данных является `ListView`. И нередко бывает важно продумать, какие инструменты использовать для управления данными в `ListView`. Одним из таких инструментов является контекстное меню. Элемент `ListView` позволяет прикрепить к каждому элементу контекстное меню через свойство **`ContextActions`**.

Итак, для работы с контекстным меню создадим новый проект. В качестве объектов с которыми будем работать, возьмем класс `Phone`

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

Далее добавим в проект модель **`PhoneViewModel`**, через которую мы будем взаимодействовать с объектами `Phone`:

```
using System.ComponentModel;

namespace MvvmApplication
{
    public class PhoneViewModel : INotifyPropertyChanged
    {
        public event PropertyChangedEventHandler PropertyChanged;
        public Phone Phone { get; set; }
        public PhonesListViewModel ListViewModel { get; set; }
        public PhoneViewModel()
        {
            Phone = new Phone();
        }
        public string Title
        {
            get { return Phone.Title; }
            set
            {
                if (Phone.Title != value)
                {
                    Phone.Title = value;
                    OnPropertyChanged("Title");
                }
            }
        }
    }
}
```

```

    }
}
}
public string Company
{
    get { return Phone.Company; }
    set
    {
        if (Phone.Company != value)
        {
            Phone.Company = value;
            OnPropertyChanged("Company");
        }
    }
}
public int Price
{
    get { return Phone.Price; }
    set
    {
        if (Phone.Price != value)
        {
            Phone.Price = value;
            OnPropertyChanged("Price");
        }
    }
}
protected void OnPropertyChanged(string propName)
{
    if (PropertyChanged != null)
        PropertyChanged(this, new PropertyChangedEventArgs(propName));
}
}
}

```

Кроме свойств-надстроек над свойствами объекта Phone здесь есть свойство, которое представляет модель PhonesListViewModel - объект списка, в котором будет находиться текущий объект PhoneViewModel. И далее добавим в проект новый класс PhonesListViewModel:

```

using System.Collections.ObjectModel;
using System.Windows.Input;
using Xamarin.Forms;

namespace MvvmApplication
{
    public class PhonesListViewModel
    {
        public ObservableCollection<PhoneViewModel> Phones { get; set; }
        public ICommand MoveToTopCommand { protected set; get; }
        public ICommand MoveToBottomCommand { protected set; get; }
        public ICommand RemoveCommand { protected set; get; }
        public PhonesListViewModel()
        {
            Phones = new ObservableCollection<PhoneViewModel>
            {
                new PhoneViewModel { Title="HP Elite z3", Price=55000, Company="HP",
                ListViewModel=this},
                new PhoneViewModel {Title="Honor 8", Price= 28000, Company="Huawei",
                ListViewModel=this},
                new PhoneViewModel {Title="iPhone SE", Price=30000, Company="Apple",
                ListViewModel=this },
                new PhoneViewModel {Title="Galaxy Note 7", Price=60000, Company="Samsung",
                ListViewModel=this },
            }
        }
    }
}

```

```

        new PhoneViewModel {Title="Lumia 950 XL", Price=36000, Company="Microsoft",
        ListViewModel=this }
    };

    MoveToTopCommand = new Command(MoveToTop);
    MoveToBottomCommand = new Command(MoveToBottom);
    RemoveCommand = new Command(Remove);
}

private void MoveToTop(object phoneObj)
{
    PhoneViewModel phone = phoneObj as PhoneViewModel;
    if (phone == null) return;
    int oldIndex = Phones.IndexOf(phone);
    if (oldIndex > 0)
        Phones.Move(oldIndex, oldIndex - 1);
}
private void MoveToBottom(object phoneObj)
{
    PhoneViewModel phone = phoneObj as PhoneViewModel;
    if (phone == null) return;
    int oldIndex = Phones.IndexOf(phone);
    if (oldIndex < Phones.Count-1)
        Phones.Move(oldIndex, oldIndex + 1);
}
private void Remove(object phoneObj)
{
    PhoneViewModel phone = phoneObj as PhoneViewModel;
    if (phone == null) return;

    Phones.Remove(phone);
}
}
}

```

Для хранения данных определена коллекция Phones, которая инициализируется в конструкторе.

Кроме этой коллекции в классе определены три команды. Команда MoveToTopCommand поднимает элемент на одну позицию вверх в списке (в реальности это выглядит как перемещение ближе к началу коллекции Phones). Команда MoveToBottomCommand, наоборот, опускает элемент на одну позицию вниз. А команда RemoveCommand удаляет элемент из списка.

В коде C# у главной страницы MainPage определим контекст данных:

```

using System;
using Xamarin.Forms;

namespace MvvmApplication
{
    public partial class MainPage : ContentPage
    {
        public MainPage()
        {
            InitializeComponent();
            this.BindingContext = new PhonesListViewModel();
        }
    }
}

```

В коде XAML определим список для вывода объектов, к каждому из которых прикрепим контекстное меню:

```

<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"

```

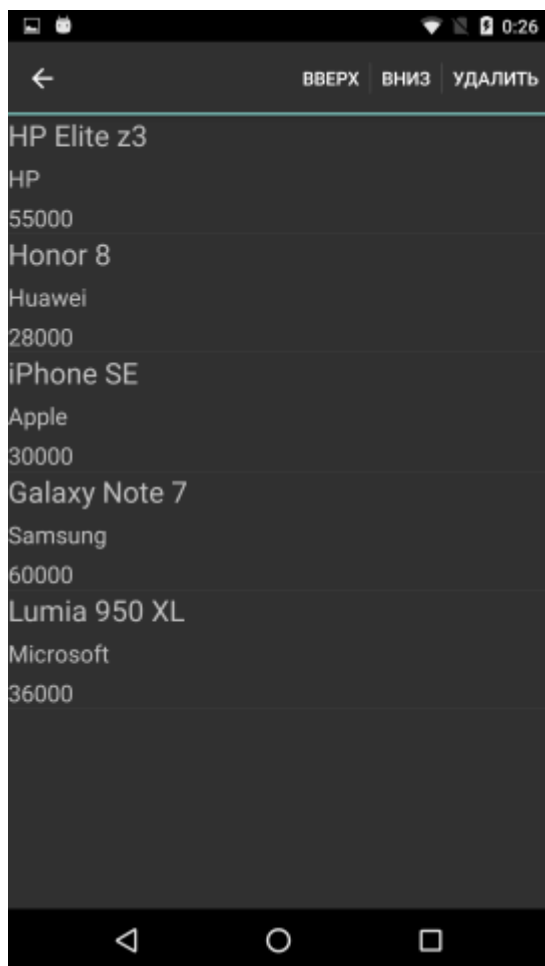
```

xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
x:Class="MvvmApplication.MainPage">
<StackLayout>
  <ListView ItemsSource="{Binding Phones}" HasUnevenRows="True">
    <ListView.ItemTemplate>
      <DataTemplate>
        <ViewCell>
          <ViewCell.ContextActions>
            <MenuItem Text="Вверх"
              Command="{Binding Path=ListViewModel.MoveToTopCommand}" CommandParameter="{Binding}" />
            <MenuItem Text="Вниз"
              Command="{Binding Path=ListViewModel.MoveToBottomCommand}" CommandParameter="{Binding}" />
            <MenuItem Text="Удалить"
              Command="{Binding Path=ListViewModel.RemoveCommand}" CommandParameter="{Binding}" />
          </ViewCell.ContextActions>
          <ViewCell.View>
            <StackLayout>
              <Label Text="{Binding Title}" FontSize="Medium" />
              <Label Text="{Binding Company}" FontSize="Small" />
              <Label Text="{Binding Price}" FontSize="Small" />
            </StackLayout>
          </ViewCell.View>
        </ViewCell>
      </DataTemplate>
    </ListView.ItemTemplate>
  </ListView>
</StackLayout>
</ContentPage>

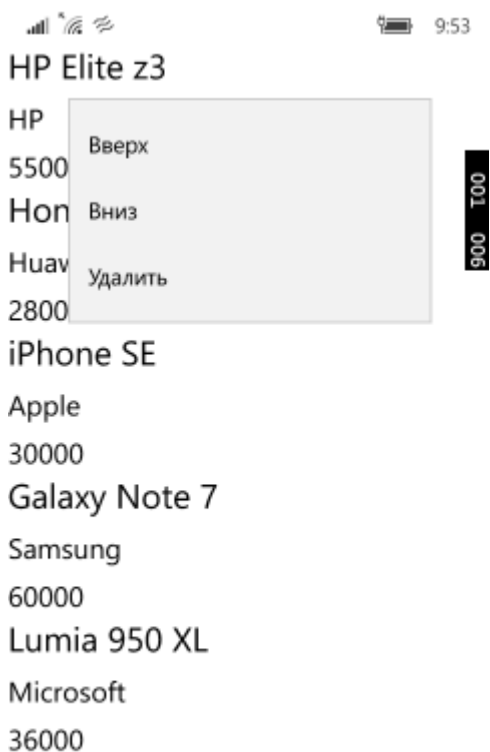
```

Для определения меню применяется свойство **ContextActions** объекта ViewCell, который представляет собой отдельный элемент. Каждый пункт в этом меню представляет элемент **MenuItem**. У MenuItem можно определить команду и параметр, который будет передаваться команде при ее выполнении. В данном примере будут вызываться команды из PhonesListViewModel, которым будет передаваться текущий объект, для которого вызвано контекстное меню.

И если мы запустим приложение и одним долгим касанием нажмем на какой-нибудь элемент списка, то отобразится контекстное меню:



При этом надо отметить, что на разных ОС (даже на разных версиях одной ОС Android) визуально контекстное меню может выглядеть иначе. Например, на Windows 10 Mobile:



Хотя здесь элементы меню используют команды и их параметры, однако нам в принципе не обязательно использовать контекстное меню именно в контексте MVVM. Так, у каждого элемента


меню есть стандартное событие Clicked, для которого мы можем определить в коде обработчик. Например:


```
<MenuItem Text="Show" Clicked="Show" />
```

А в коде прописать какой-нибудь метод обработки:

```
private async void Show(object sender, EventArgs e)
{
    await DisplayAlert("Контекстное меню", "Пункт Show", "OK");
}
```


В тоже время в MVVM использование контекстного меню имеет свои плюсы, в частности, возможность передачи параметра.





## Конструктор Platforma LP

Запусти сайт за час. И  
подключи виджеты в  
два клика. Выгода 10%  
при оплате 3-х мес.



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

[G+](#)

Яндекс.Директ

Тачка DGM GT-1081

21vek.by

Яндекс.Директ

Профильная труба  
для столбов

metagarant-minsk.by

11 Комментариев metanit.com

 Войти ▾

 Рекомендовать  Поделиться

Лучшее в начале ▾



Присоединиться к обсуждению...

ВОЙТИ С ПОМОЩЬЮ

ИЛИ ЧЕРЕЗ DISQUS 

Имя



**Бато Дышенов** • год назад

Подскажите пожалуйста как добавить контекстное меню в другие визуальные объекты типа WebView и Editor

^ | v • Ответить • Поделиться ›



**Metanit** Модератор ➔ Бато Дышенов • год назад

ну если только создать свое контекстное меню

^ | v • Ответить • Поделиться ›



**Бато Дышенов** ➔ Metanit • год назад

Это получается нужно переопределять рендер ? Моя задача заключается в том что бы добавить еще одну строку в контекстное меню которое идет по умолчанию

там в облаках перед народом

Через леса, через моря

Колдунья гатыря;

В темноте древна тужит,

А бурый волк ей верно служит;

Там ступа с Бабою Ягой

Идёт, бредёт сама собой,

Там царь Кащей над златом чахнет;

^ | v • Ответить • Поделиться ›



**Metanit** Модератор ➔ Бато Дышенов • год назад

да переопределяйте рендер

^ | v • Ответить • Поделиться ›



**Бато Дышенов** ➔ Metanit • год назад

А по какому событию оно должно появляться

^ | v • Ответить • Поделиться ›



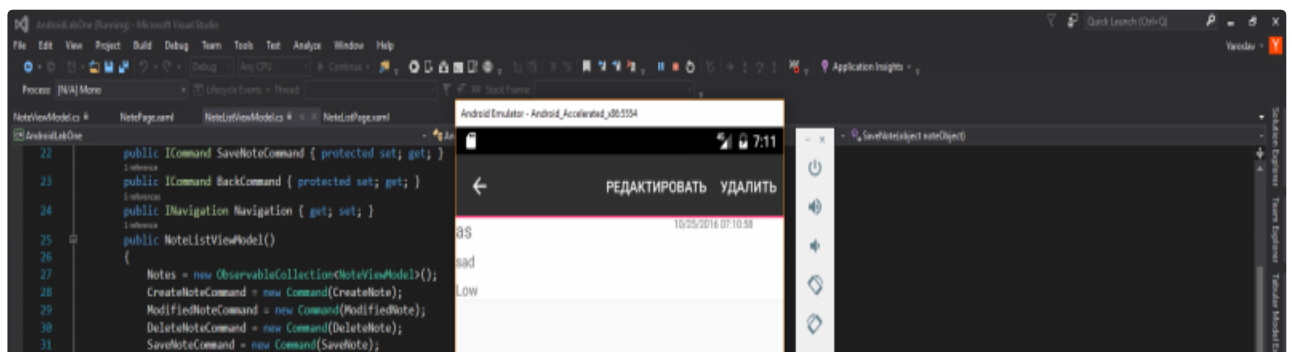
**Metanit** Модератор ➔ Бато Дышенов • год назад

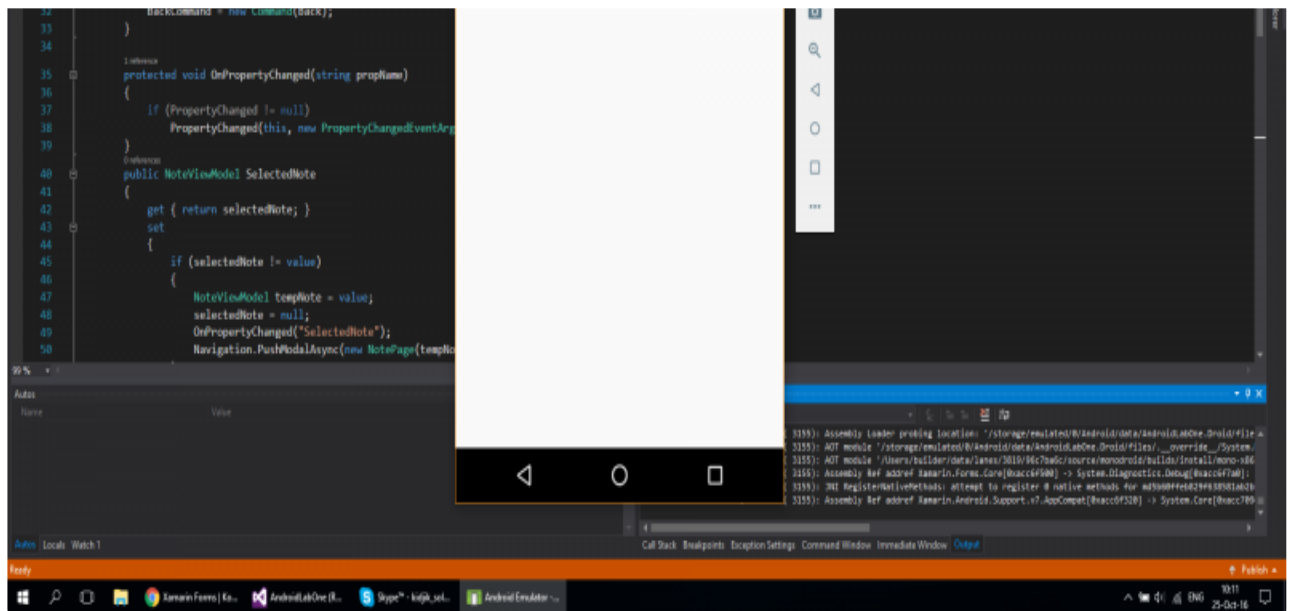
по долгому нажатию на элемент

^ | v • Ответить • Поделиться ›



**Ярослав Орлов** • год назад





Вот еще вопрос, как настроить контекстное меню как у Вас, ибо как видите оно немного видоизменено по умолчанию.

^ | ▾ • Ответить • Поделиться ›



**Metanit** Модератор ➔ Ярослав Орлов • год назад

внешний вид меню зависит от платформы и даже подверсий в рамках одной платформы, поэтому тут вряд ли что-то можно сделать

^ | ▾ • Ответить • Поделиться ›



**Ярослав Орлов** • год назад

```
<stacklayout>
<button text="Добавить" command="{Binding CreateNoteCommand}"/>
<listview x:name="notesList" itemsSource="{Binding Notes}" selecteditem="{Binding
SelectedNote, Mode=TwoWay}" hasunevenrows="True">
<listview.itemtemplate>
<datatemplate>
<viewcell>
<viewcell.contextactions>
<menuitem text="Редактировать" command="{Binding ModifiedNoteCommand}"
commandparameter="{Binding}"/>
<menuitem text="Удалить" command="{Binding DeleteNoteCommand}" commandparameter="{
Binding}"/>
</viewcell.contextactions>
<viewcell.view>
<stacklayout>
<stacklayout orientation="Horizontal">
<label text="{Binding Name}" fontsize="Medium"/>
<label text="{Binding NoteDate}" fontsize="Micro" margin="230, 0, 0, 0"/>
```

показать больше

^ | ▾ • Ответить • Поделиться ›



**Metanit** Модератор ➔ Ярослав Орлов • год назад

а ModifiedNoteCommand и другая команда определены во ViewModel или в элементе, который выводится во ViewCell (скорее всего не там определены



элементе, который выводится во viewcell (скорее всего не там определена команда)

^ | v • Ответить • Поделиться ›



**Ярослав Орлов** → Metanit • год назад

Слушайте, Вы просто гений, спасибо большое и правда не в том View была команда.

^ | v • Ответить • Поделиться ›

ТАКЖЕ НА METANIT.COM

## Angular в ASP.NET Core | CRUD и маршрутизация. Часть 1

1 комментарий • 3 месяца назад



**Vadim Prokopchuk** — Добрый вечер. Попытался в текущий пример добавить стилизацию, но столкнулся с

## Go | Соответствие интерфейсу

5 комментариев • 2 месяца назад



**Metanit** — менять необязательно

## C++ | Шаблоны функций


2 комментариев • 3 месяца назад




**Владимир** — Присоединяюсь к вопросу. Хотелось бы урок по многопоточности.

## Kotlin | Введение в язык. Первая программа

1 комментарий • 2 месяца назад





Всё для чистоты  
и гигиены в организациях.  
Доступные цены.  
Быстрая доставка.

[Подробнее](#)