

Министерство образования Новосибирской области

ГБПОУ НСО «Новосибирский авиационный технический колледж им. Б.С. Галушака»

УТВЕРЖДАЮ:
Председатель цикловой
комиссии УГС 09.00.00
___/ А.А. Терехова
Протокол № ___ от «___»
_____ 20__ г.

УТВЕРЖДАЮ:
Председатель цикловой
комиссии УГС 09.00.00
___/ А.А. Терехова
Протокол № ___ от «___»
_____ 20__ г.

УТВЕРЖДАЮ:
Председатель цикловой
комиссии УГС 09.00.00
___/ А.А. Терехова
Протокол № ___ от «___»
_____ 20__ г.

**List View. Обработка данных для приложений с графическим
пользовательским интерфейсом**

Методические указания к практическому занятию № ___

Профессиональный модуль: ПМ.01 Разработка модулей программного обеспечения для
компьютерных систем

Междисциплинарный курс: МДК 01.01 Разработка программных модулей

Специальность: 09.02.07 Информационные системы и программирование

Разработал:
А.С. Иванов

Содержание

1 Теоретический раздел	3
1.1 Основные понятия и термины	3
1.2 Вёрстка элемента ListView	5
2 Практический раздел	9
2.1 Использование ListView в программных решениях для бизнеса	9

1 Теоретический раздел

1.1 Основные понятия и термины

Класс `ListView` — это специализированный класс спискового типа, предназначенный для отображения различных представлений одних и тех же данных. Он особенно удобен, когда требуется создать представление, состоящее из множества столбцов и отображающее о каждом элементе данных несколько различных фрагментов информации.

Класс `ListView` унаследован от `ListBox` и дополняет его одной единственной деталью: свойством `View`. Свойство `View` представляет собой еще одну точку для создания многофункциональных списковых представлений. Если это свойство не установлено, элемент `ListView` ведет себя просто точно так же, как и его менее мощный предок — класс `ListBox`. Однако он становится гораздо более интересным, когда разработчик предоставляет объект представления, указывающий, каким должно быть форматирование и стили у элементов данных.

На рисунке 1 представлены графические представления `ListBox` и `ListView`

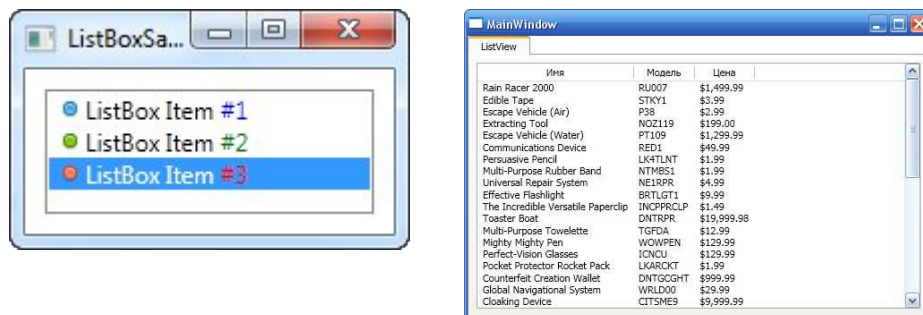


Рисунок 1 – Графические представления элементов `ListBox` и `ListView`

Формально свойство `View` указывает на экземпляр любого класса, унаследованного от `ViewBase` (который представляет собой абстрактный класс). Класс `ViewBase` удивительно прост: в действительности это не более чем оболочка, объединяющая вместе два стиля. Один из этих стилей применяется к элементу управления `ListView` (и указывается в свойстве `DefaultStyleKey`), в то время как другой применяется элементам внутри `ListView` (и указывается в свойстве `ItemContainerDefaultStyleKey`). Свойства `DefaultStyleKey` и `ItemContainerDefaultStyleKey` фактически стиль не предоставляют, а вместо этого они просто возвращают указывающий на него объект `ResourceKey`.

Также, свойство `View` является полезной абстракцией, предоставляющая разработчику следующие преимущества:

- представления, пригодные для многократного применения. ListView выделяет все касающиеся представления детали в один объект. Это упрощает создание представлений, не зависящих от данных и пригодных для применения с более чем одним списком;
- множественные представления. Отделение элемента управления ListView от объектов View также упрощает переключение между множеством различных представлений в одном и том же списке. (Например, именно такая технология применяется в проводнике Windows для получения различных представлений файлов и папок пользователя.) Точно такую же функциональную возможность можно создать и за счет динамического изменения шаблонов и стилей, но построить единственный объект, инкапсулирующий все детали представления, все-таки гораздо легче;
- более удобная организация. Объект представления упаковывает два стиля: один для корневого элемента управления ListView и еще один — для отдельных элементов в списке. Поскольку эти стили упаковываются вместе, очевидно, что эти два фрагмента связаны между собой и могут разделять определенные детали и взаимозависимости. Например, для основанного на столбцах элемента управления ListView это имеет большой смысл, поскольку ему необходимо, чтобы размещение заголовков столбцов совпадало с размещением предназначенных для этих столбцов данных.\

Применение этой модели открывает замечательные возможности для создания ряда полезных готовых представлений, которыми бы могли пользоваться все разработчики. К сожалению, в настоящее время в состав WPF входит пока только один объект представления — объект GridView. Хотя его можно и удобно использовать для построения списков с множеством столбцов, при наличии каких-то других потребностей придется создавать собственное специальное представление. В последующих разделах будет показано, как делать то и другое.

Элемент GridView будет замечательным выбором, если нужно, чтобы отображаемые данные можно было конфигурировать, а представление в стиле сетки было одной из доступных для пользователя опций. Но если требуется, чтобы в сетке дополнительно поддерживались расширенная стилизация, выбор или редактирование, придется перейти к использованию полнофункционального элемента управления DataGrid.

1.2 Вёрстка элемента ListView

GridView — это класс, который наследуется от ViewBase и предоставляет списковое представление с множеством столбцов. Определяются эти столбцы путем добавления в коллекцию GridViewColumns объектов GridViewColumn.

GridView и GridViewColumn предлагают небольшой набор полезных методов, которые разработчик может использовать для настройки внешнего вида своего списка. Чтобы создать самый простой прямолинейный список (вроде Details (Список) в проводнике Windows), потребуется установить для каждого объекта GridViewColumn всего лишь два свойства: Header и DisplayMemberBinding. Свойство Header отвечает за размещаемый в верхней части столбца текст, а свойство DisplayMemberBinding содержит привязку, извлекающую из каждого элемента данных подлежащий отображению фрагмент информации.

Ниже приведен код разметки, необходимый для определения трех используемых в этом примере столбцов:

```
<ListView Name="lstw" Margin="5">
    <ListView.View>
        <GridView>
            <GridView.Columns>
                <GridViewColumn Header="Имя"
DisplayMemberBinding="{Binding Path=ModelName}"></GridViewColumn>
                <GridViewColumn Header="Модель"
DisplayMemberBinding="{Binding Path=ModelNumber}"></GridViewColumn>
                <GridViewColumn Header="Цена"
DisplayMemberBinding="{Binding Path=UnitCost,
StringFormat={}{0:C}}"></GridViewColumn>
            </GridView.Columns>
        </GridView>
    </ListView.View>
</ListView>
```

В результате имеем следующей список с данными на рисунке 2.



Имя	Модель	Цена
Rain Racer 2000	RU007	\$1,499.99
Edible Tape	STKY1	\$3.99
Escape Vehicle (Air)	P38	\$2.99
Extracting Tool	NOZ119	\$199.00
Escape Vehicle (Water)	PT109	\$1,299.99
Communications Device	RED1	\$49.99
Persuasive Pencil	LK4TLNT	\$1.99
Multi-Purpose Rubber Band	NTMBS1	\$1.99
Universal Repair System	NE1RPR	\$4.99
Effective Flashlight	BRTLGT1	\$9.99
The Incredible Versatile Paperclip	INCPPRCLP	\$1.49
Toaster Boat	DNTRPR	\$19,999.98
Multi-Purpose Towelette	TGFDA	\$12.99
Mighty Mighty Pen	WOWPEN	\$129.99
Perfect-Vision Glasses	ICNCU	\$129.99
Pocket Protector Rocket Pack	LKARCKT	\$1.99
Counterfeit Creation Wallet	DNTGCGHT	\$999.99
Global Navigational System	WRLD00	\$29.99
Cloaking Device	CITSME9	\$9,999.99

Рисунок 2 – Вид ListView

Замечание: данные, представленные в этом элементе, не указаны в коде разметки. Разметке включается в себя обращение к элементам наполнения.

Действительно, данные, служащие наполнителем интерактивных элементов, берутся из базы данных. В классических оконных WPF-приложениях база данных импортируется в решение как модель при помощи ORM-решения EntityFramework.

Как правило, создается некоторый класс данных, принимающий данные из модели. Пример такого класса представлен ниже:

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

В данном случае мы определяем класс Phone, который хранит данные о заголовке, производителе и цене телефона. Это минимальный набор для карточки товара, встречающихся, например, в приложении магазина по продаже компьютерной техники.

Теперь, при вёрстке интерактивных элементов, можно обратиться к данным посредством привязки.

Ниже приведён пример такой разметки.

```
<Window x:Class="ControlsApp.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-
```

```

compatibility/2006"
    xmlns:local="clr-namespace:ControlsApp"
    xmlns:col="clr-namespace:System.Collections;assembly=mscorlib"
    mc:Ignorable="d"
    Title="ListView" Height="220" Width="300">
<Grid Background="Lavender">
    <ListView Name="phonesList" ItemsSource="{DynamicResource
ResourceKey=phones}" >
        <ListView.View>
            <GridView>
                <GridViewColumn DisplayMemberBinding="{Binding
Path=Title}">Модель</GridViewColumn>
                <GridViewColumn DisplayMemberBinding="{Binding
Path=Company}" Width="100">Компания</GridViewColumn>
                <GridViewColumn DisplayMemberBinding="{Binding
Path=Price}">Цена</GridViewColumn>
            </GridView>
        </ListView.View>
        <ListView.Resources>
            <col:ArrayList x:Key="phones">
                <local:Phone Title="iPhone 6S" Company="Apple"
Price="54990" />
                <local:Phone Title="Lumia 950" Company="Microsoft"
Price="39990" />
                <local:Phone Title="Nexus 5X" Company="Google"
Price="29990" />
                <local:Phone Title = "Galaxy Edge" Company = "Samsung"
Price = "45670" />
            </col:ArrayList>
        </ListView.Resources>
    </ListView>
</Grid>
</Window>

```

В результате получим некоторый список данных, как на рисунке 3.



The screenshot shows a standard Java Swing window titled "ListView". Inside the window is a table with three columns: "Модель" (Model), "Компания" (Company), and "Цена" (Price). The table contains four rows of data representing different smartphone models and their prices.

Модель	Компания	Цена
iPhone 6S	Apple	54990
Lumia 950	Microsoft	39990
Nexus 5X	Google	29990
Galaxy Edge	Samsung	45670

Рисунок 3 – ListView, наполненный переданными в него данными при помощи привязки

2 Практический раздел

2.1 Использование ListView в программных решениях для бизнеса

Предположим, что разрабатывается программное решение для автосервиса. Пользователь, при входе в приложение, должен понимать, какие товары предоставляет для покупки автосервис. В базе данных есть таблица, связанная с товарами. Поля таблицы представлены на рисунке 4.

Product			
	Column Name	Data Type	Allow Nulls
🔑	ProductArticleNumber	nvarchar(100)	<input type="checkbox"/>
	ProductName	nvarchar(MAX)	<input type="checkbox"/>
	ProductDescription	nvarchar(MAX)	<input type="checkbox"/>
	ProductCategory	nvarchar(MAX)	<input type="checkbox"/>
	ProductImage	nvarchar(MAX)	<input type="checkbox"/>
	ProductManufacturer	nvarchar(MAX)	<input type="checkbox"/>
	ProductCost	decimal(19, 4)	<input type="checkbox"/>
	ProductDiscountAmount	tinyint	<input checked="" type="checkbox"/>
	ProductQuantityInStock	int	<input type="checkbox"/>
	ProductStatus	nvarchar(MAX)	<input type="checkbox"/>
	Unit	nvarchar(50)	<input type="checkbox"/>
	MaxDiscountAmount	tinyint	<input type="checkbox"/>
	Supplier	nvarchar(50)	<input type="checkbox"/>
	CountInPack	int	<input checked="" type="checkbox"/>
	MinCount	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Рисунок 4 – Поля таблицы товаров автосервиса

В классическое WPF-решение мигрирована таблица в комплекте со всей базой данных через EntityFramework. См. рисунок 5.

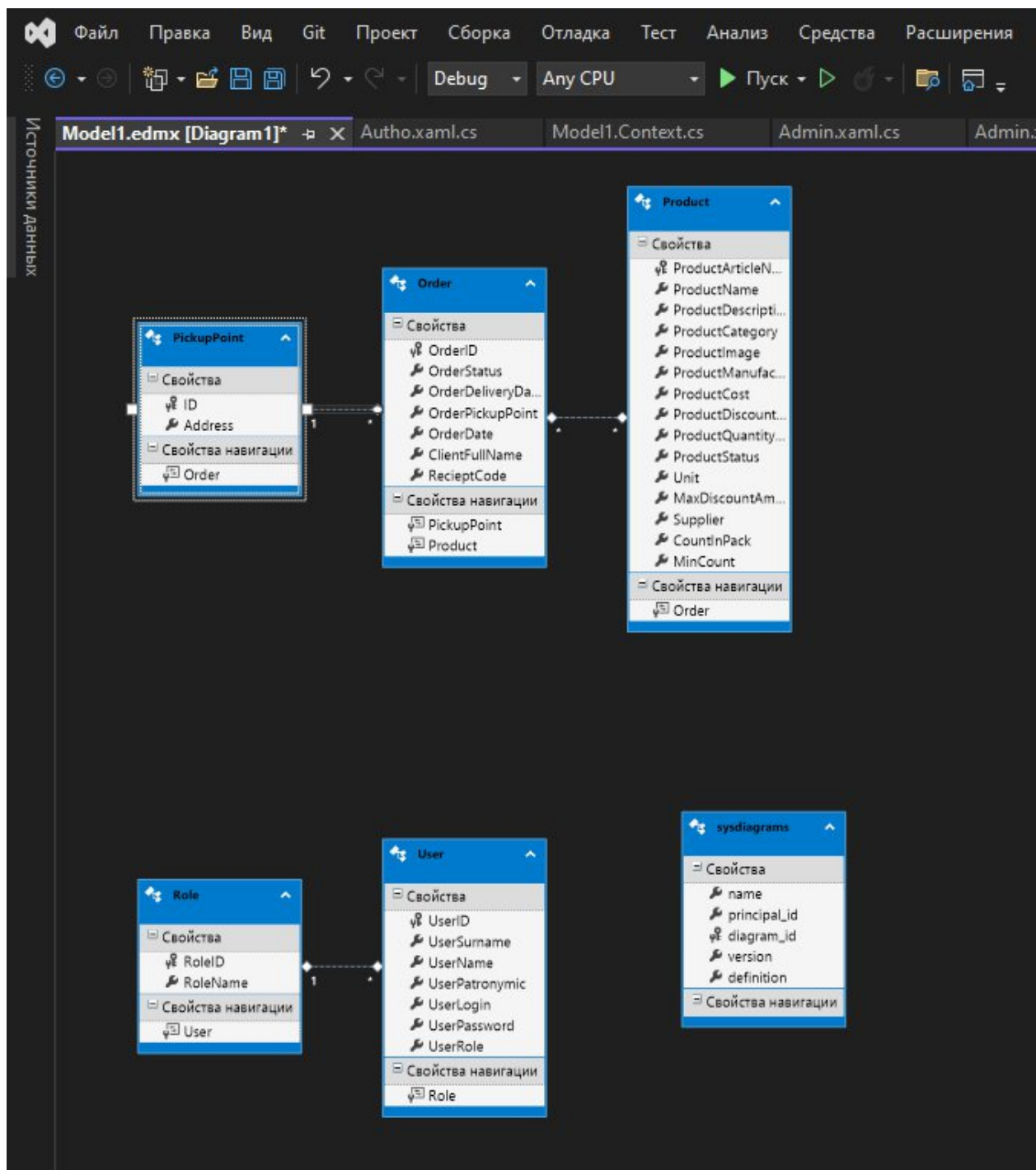


Рисунок 5 – EDM-модель базы данных

Соответственно, при миграции данных, EntityFramework определяет классы каждой сущности, находящихся в модели. Класс товаров выглядит следующим образом:

```
//-----
//
// <auto-generated>
//     Этот код создан по шаблону.
//
//     Изменения, вносимые в этот файл вручную, могут привести к
//     непредвиденной работе приложения.
//     Изменения, вносимые в этот файл вручную, будут перезаписаны при
//     повторном создании кода.
```

```
// </auto-generated>
//-----
----

namespace rul.Entities
{
    using System;
    using System.Collections.Generic;

    public partial class Product
    {
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage",
"CA2214:DoNotCallOverridableMethodsInConstructors")]
        public Product()
        {
            this.Order = new HashSet<Order>();
        }

        public string ProductArticleNumber { get; set; }
        public string ProductName { get; set; }
        public string ProductDescription { get; set; }
        public string ProductCategory { get; set; }
        public string ProductImage { get; set; }
        public string ProductManufacturer { get; set; }
        public decimal ProductCost { get; set; }
        public Nullable<byte> ProductDiscountAmount { get; set; }
        public int ProductQuantityInStock { get; set; }
        public string ProductStatus { get; set; }
        public string Unit { get; set; }
        public byte MaxDiscountAmount { get; set; }
        public string Supplier { get; set; }
        public Nullable<int> CountInPack { get; set; }
        public Nullable<int> MinCount { get; set; }

        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage",
"CA2227:CollectionPropertiesShouldBeReadOnly")]
        public virtual ICollection<Order> Order { get; set; }
    }
}
```

Для отображения товаров пользователям приложения необходима страница клиента. Выглядит она следующим образом:

The screenshot shows a web application interface for displaying products. The interface is structured with a grid system, indicated by blue lines and dimensions. The header area has a width of 200 units and a height of 50 units. Below the header, there are three main columns, each with a width of 200 units, and a final column with a width of 80 units. The product cards are arranged in a grid, with each card having a width of 200 units and a height of 100 units. The product cards display the following information: ProductName, ProductDescription, ProductManufacturer, ProductCost, and ProductDiscountAmount. The layout is clean and modern, with a white background and blue accents.

Рисунок 6 – Страница клиента

Основное наполнение страницы будет осуществляться с помощью элемента ListView и настройки скроллинга страницы.

Вёрстка страницы:

```
<Page x:Class="rul.Pages.Client"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:rul.Pages" xmlns:sys="clr-
namespace:System;assembly=mscorlib"
      mc:Ignorable="d"
      d:DesignHeight="450" d:DesignWidth="800"
      Title="Client">
    <Page.Resources>
      <sys:String x:Key="defaultImage">
        pack://application:,,,/Resources/picture.png
      </sys:String>
    </Page.Resources>

    <Grid>
      <Grid.ColumnDefinitions>
        <ColumnDefinition Width="200"/>
        <ColumnDefinition Width="200"/>
        <ColumnDefinition Width="200"/>
        <ColumnDefinition Width="80"/>
      </Grid.ColumnDefinitions>
      <Grid.RowDefinitions>
        <RowDefinition Height="30"/>
        <RowDefinition Height="56"/>
        <RowDefinition />
        <RowDefinition Height="60"/>
      </Grid.RowDefinitions>

      <StackPanel Orientation="Horizontal" Grid.Row="0" Grid.Column="2"
Grid.ColumnSpan="3" HorizontalAlignment="Right">
        <TextBlock Text="Пользователь"/>
        <TextBlock x:Name="txtFullName"/>
      </StackPanel>

      <TextBox Name ="txtSearch" Grid.Row="1" Grid.Column="0"
VerticalAlignment="Center" Margin="5, 0, 5, 0"
SelectionChanged="txtSearch_SelectionChanged"/>
      <ComboBox Name ="cmbSorting" Grid.Row="1" Grid.Column="1"
SelectedIndex="0" VerticalAlignment="Center" Margin="5, 0, 5, 0"
ItemsSource="{Binding SortingList}"
SelectionChanged="cmbSorting_SelectionChanged"/>
      <ComboBox Name ="cmbFilter" Grid.Row="1" Grid.Column="2"
SelectedIndex="0" VerticalAlignment="Center" Margin="5, 0, 5, 0"
ItemsSource="{Binding FilterList}"
SelectionChanged="cmbFilter_SelectionChanged"/>

      <StackPanel Orientation="Horizontal" Grid.Row="1" Grid.Column="3"
VerticalAlignment="Center" HorizontalAlignment="Center">
        <TextBlock x:Name="txtResultAmount"/>
        <TextBlock Text="Из"/>
        <TextBlock x:Name="txtAllAmount"/>
      </StackPanel>

      <ListView Name="LViewProduct" Grid.Column="0" Grid.ColumnSpan="4"
Grid.Row="2" Margin="5,5,5,5" d:ItemsSource="{d:SampleData ItemCount=1}">
```

```

        <ListView.ItemContainerStyle>
            <Style TargetType="ListViewItem">
                <Setter Property="HorizontalContentAlignment"
Value="Stretch"/>
            </Style>
        </ListView.ItemContainerStyle>
        <ListView.ItemTemplate>
            <DataTemplate>
                <Border BorderBrush="Black" BorderThickness="1"
Background="{Binding Background}">
                    <Grid>
                        <Grid.ColumnDefinitions>
                            <ColumnDefinition Width="200"/>
                            <ColumnDefinition Width="*" />
                            <ColumnDefinition Width="200"/>
                        </Grid.ColumnDefinitions>

                        <Image Width="100" Height="100" Margin="5"
Source="{Binding ImgPath, FallbackValue={StaticResource defaultImage}}"/>

                        <StackPanel Grid.Column="1" Margin="5">
                            <TextBox Text="{Binding ProductName}"
FontWeight="Bold"/>
                            <TextBox Text="{Binding
ProductDescription}"/>
                            <TextBox Text="{Binding
ProductManufacture}"/>
                            <StackPanel Orientation="Horizontal">
                                <TextBox Text="{Binding ProducCost}"
TextDecorations="Strikethrough"/>
                                <TextBox Text="{Binding
CostWithDiscount, StringFormat='\0 рублей'}" FontWeight="Bold"/>
                            </StackPanel>
                        </StackPanel>

                        <TextBox Grid.Column="2" Text="{Binding
ProductDiscountAmount, StringFormat='\0 %'}" VerticalAlignment="Center"/>
                    </Grid>
                </Border>
            </DataTemplate>
        </ListView.ItemTemplate>
        <ListView.ContextMenu>
            <ContextMenu Name="contextMenu">
                <MenuItem Name="btnAddProduct" Header="Добавить к
заказу" Click="btnAddProduct_Click"/>
            </ContextMenu>
        </ListView.ContextMenu>
    </ListView>
    <Button x:Name="btnOrder" Grid.Row="3" Grid.Column="0"
Content="Заказ" Visibility="Collapsed" HorizontalAlignment="Left"
Width="200" Margin="20, 0, 0, 0" Click="btnOrder_Click"/>
</Grid>
</Page>

```

Пояснения:

Для отображения изображений товаров имеется настройка `<Image Width="100" Height="100" Margin="5" Source="{Binding ImgPath, FallbackValue={StaticResource defaultImage}}"/>` , получающая информацию из

каталога ресурсов. При отсутствии изображения товара будет отображаться крестик, означающий отсутствие изображения, созданный с помощью надстройки вёрстки

```
<Page.Resources>
    <sys:String x:Key="defaultImage">
        pack://application:,,,/Resources/picture.png
    </sys:String>
</Page.Resources>
```

Далее идет основной компонент наполнения – **ListView**:

```
<ListView Name="LViewProduct" Grid.Column="0" Grid.ColumnSpan="4"
Grid.Row="2" Margin="5,5,5,5" d:ItemsSource="{d:SampleData ItemCount=1}">
    <ListView.ItemContainerStyle>
        <Style TargetType="ListViewItem">
            <Setter Property="HorizontalContentAlignment" Value="Stretch"/>
        </Style>
    </ListView.ItemContainerStyle>
    <ListView.ItemTemplate>
        <DataTemplate>
            <Border BorderBrush="Black" BorderThickness="1"
Background="{Binding Background}">
                <Grid>
                    <Grid.ColumnDefinitions>
                        <ColumnDefinition Width="200"/>
                        <ColumnDefinition Width="*/>
                        <ColumnDefinition Width="200"/>
                    </Grid.ColumnDefinitions>

                    <Image Width="100" Height="100" Margin="5"
Source="{Binding ImgPath, FallbackValue={StaticResource defaultImage}}"/>

                    <StackPanel Grid.Column="1" Margin="5">
                        <TextBox Text="{Binding ProductName}"
FontWeight="Bold"/>
                        <TextBox Text="{Binding ProductDescription}"/>
                        <TextBox Text="{Binding ProductManufacture}"/>
                        <StackPanel Orientation="Horizontal">
                            <TextBox Text="{Binding ProducCost}"
TextDecorations="Strikethrough"/>
                            <TextBox Text="{Binding CostWithDiscount,
StringFormat='\0 рублей'}" FontWeight="Bold"/>
                        </StackPanel>
                    </StackPanel>

                    <TextBox Grid.Column="2" Text="{Binding
ProductDiscountAmount, StringFormat='\0 %'}" VerticalAlignment="Center"/>
                </Grid>
            </Border>
        </DataTemplate>
    </ListView.ItemTemplate>
    <ListView.ContextMenu>
        <ContextMenu Name="contextMenu">
            <MenuItem Name="btnAddProduct" Header="Добавить к заказу"
Click="btnAddProduct_Click"/>
        </ContextMenu>
    </ListView.ContextMenu>
</ListView>
```

Вёрстка разбита на несколько частей.

- **ListView:**
`Name="LViewProduct"`: Имя элемента, которое можно использовать для доступа к нему в коде.
`Grid.Column="0" Grid.ColumnSpan="4" Grid.Row="2" Margin="5,5,5,5"`: Определяет положение и отступы элемента в сетке.
`d:ItemsSource="{d:SampleData ItemCount=1}"`: Используется для отображения примеров данных в режиме дизайна.
- **ListView.ItemContainerStyle:**
`Style TargetType="ListViewItem"`: Определяет стиль для элементов списка.
`Setter Property="HorizontalContentAlignment" Value="Stretch"`: Устанавливает горизонтальное выравнивание содержимого элементов списка на "Stretch", что позволяет содержимому заполнять всю доступную ширину.
- **ListView.ItemTemplate:**
`DataTemplate`: Определяет шаблон для отображения каждого элемента списка.
`Border`: Определяет границу и фон для каждого элемента списка.
`BorderBrush="Black" BorderThickness="1" Background="{Binding Background}"`: Устанавливает чёрную границу толщиной 1 и привязывает фон к свойству `Background` данных.
- **Grid:**
`Grid.ColumnDefinitions`: Определяет три столбца для сетки.
Первый столбец шириной 200.
Второй столбец занимает оставшееся пространство.
Третий столбец шириной 200.
- **Image:**
`Width="100" Height="100" Margin="5" Source="{Binding ImgPath, FallbackValue={StaticResource defaultImage}}"`: Отображает изображение продукта. Источник изображения привязывается к свойству `ImgPath`, а если значение отсутствует, используется ресурс `defaultImage`.
- **StackPanel:**
`Grid.Column="1" Margin="5"`: Определяет панель для отображения текстовых полей во втором столбце.
`TextBox`: Отображает различные свойства продукта:
`ProductName`: Название продукта.
`ProductDescription`: Описание продукта.
`ProductManufacture`: Производитель продукта.

ProducCost: Стоимость продукта с зачёркиванием.

CostWithDiscount: Стоимость продукта со скидкой, форматированная как "0 рублей".

– TextBox:

Grid.Column="2" Text="{Binding ProductDiscountAmount, StringFormat='\0 %'}"
VerticalAlignment="Center": Отображает размер скидки в третьем столбце, форматированный как "0 %".

– ListView.ContextMenu:

ContextMenu Name="contextMenu": Определяет контекстное меню для элементов списка.

MenuItem Name="btnAddProduct" Header="Добавить к заказу"

Click="btnAddProduct_Click": Определяет элемент контекстного меню для добавления продукта к заказу. При нажатии вызывается обработчик события btnAddProduct_Click.

Для получения окончательного результата применим реализацию выборки данных из базы:

```
public partial class MainWindow : Window
{
    public MainWindow()
    {
        InitializeComponent();
        LoadProducts();
    }

    private void LoadProducts()
    {
        var products = GetProductsFromDatabase();
        LViewProduct.ItemsSource = products;
    }

    private void btnAddProduct_Click(object sender,
    RoutedEventArgs e)
    {
        // Обработчик события для добавления продукта к заказу
        if (LViewProduct.SelectedItem is Product
        selectedProduct)
        {
            // Логика добавления продукта к заказу
        }
    }
}
```



```
        MessageBox.Show($"Добавлен продукт: {selectedProduct.ProductName}");
    }
}
```

Результат:

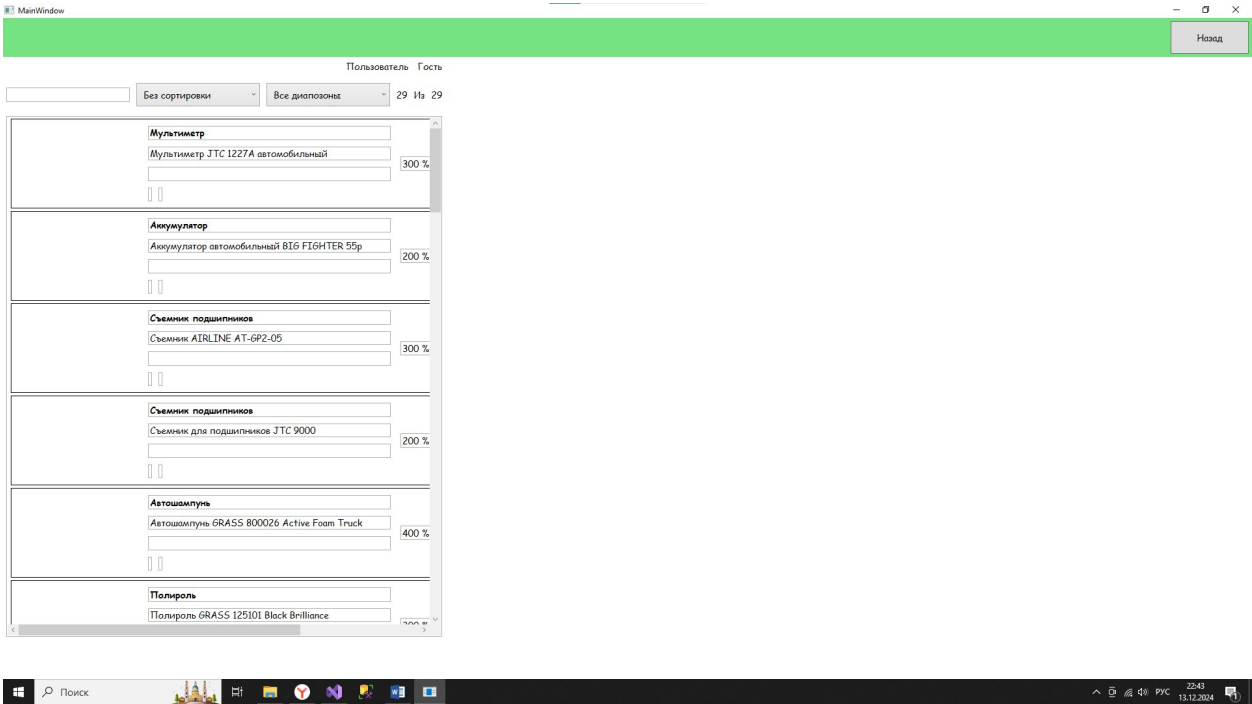


Рисунок 7 – Результат отображения данных

Данные, предназначенные для наполнения, находятся в базе данных

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The left pane displays the 'Object Explorer' with a tree view of the database structure. The right pane shows a query window with the following SQL query:

```
SELECT TOP (1000) [ProductArticleNumber],
[ProductName],
[ProductDescription],
[ProductCategory],
[ProductImage],
[ProductManufacturer],
[ProductCost],
[ProductDiscountAmount],
[ProductQuantityInStock],
[ProductStatus],
[Unit],
[MaxDiscountAmount],
[Supplier],
[CountInPack],
[Discount]
FROM [Trade].[dbo].[Product]
```

The query results are displayed in a table with 29 rows. The columns are: ProductArticleNumber, ProductName, ProductDescription, ProductCategory, ProductImage, ProductManufacturer, ProductCost, ProductDiscountAmount, ProductQuantityInStock, ProductStatus, Unit, MaxDiscountAmount, Supplier, and CountInPack. The data includes various automotive and electronic components.

ProductArticleNumber	ProductName	ProductDescription	ProductCategory	ProductImage	ProductManufacturer	ProductCost	ProductDiscountAmount	ProductQuantityInStock	ProductStatus	Unit	MaxDiscountAmount	Supplier	CountInPack
1	M56277	Мультиметр	Мультиметр JTC 1227A автомобильный	Аксессуары	JTC	14200.0000	3	12		шт.	5	Максидом	NULL
2	A76284	Аккумулятор	Аккумулятор автомобильный BIG FIGHTER 55p	Автозапчасти	A76284.jpg	4500.0000	2	24		шт.	30	Максидом	NULL
3	D378E4	Съемник подшипников	Съемник AIRLINE AT-0P2-05	Съемники подшипников	D378E4.jpg	1400.0000	3	2		шт.	20	220-volt	NULL
4	D79976	Съемник подшипников	Съемник для подшипников JTC 9000	Съемники подшипников	JTC	1800.0000	2	6		шт.	25	220-volt	NULL
5	E679P3	Автошланг	Автошланг GRASS 800026 Active Foam Truck	Автосервис	GRASS	4000.0000	4	14		шт.	15	220-volt	NULL
6	E932T8	Полтерьер	Полтерьер GRASS 125101 Black Balance	Автосервис	GRASS	2100.0000	3	23		шт.	25	220-volt	NULL
7	F04894	Антифриз	Антифриз синтетический MOBIL ANTIFREEZE EXTRA	Автосервис	MOBIL	530.0000	2	13		шт.	15	220-volt	NULL
8	F938T5	Антифриз	Антифриз красный TOL LLC21212	Автосервис	TOL	1200.0000	4	34		шт.	15	220-volt	NULL
9	H572T6	Парктроник	Парктроник AIRLINE APS-BL-02	Автозапчасти	AIRLINE	2900.0000	5	12		шт.	15	220-volt	NULL
10	K702L6	Ключ	Ключ JONNESWAY W233032 (30 / 32 мм)	Рулевые инструменты	JONNESWAY	1600.0000	3	9		шт.	15	220-volt	NULL
11	K830R4	Котак для колеса	Котак для колеса AIRLINE Супер Астра R16 универс.	Автозапчасти	K830R4.jpg	915.0000	3	14		шт.	20	Максидом	NULL
12	K948L6	Набор ключей	Набор ключей наконечник STV 00-00010990 Sup.	Рулевые инструменты	K948L6.jpg	780.0000	2	23		шт.	15	220-volt	NULL
13	L802Y5	Полата	Полата саперная AIRLINE AB-S-03	Аксессуары	AIRLINE	870.0000	4	23		шт.	5	220-volt	NULL
14	O393R4	Отвертка	Отвертка JONNESWAY D04P2100	Рулевые инструменты	O393R4.jpg	460.0000	3	14		шт.	15	220-volt	NULL
15	P023T2	Провода прикуривания	Провода прикуривания в сумке SMART CABLE 700 4.5m	Автозапчасти	SMART	3400.0000	2	6		шт.	20	Максидом	NULL
16	P307T5	Провода прикуривания	Провода прикуривания в сумке EXPERT 400A 2.5m	Автозапчасти	EXPERT	700.0000	4	2		шт.	10	Максидом	NULL
17	P439R4	Пассатижи	Пассатижи HAMMER Рав 601-050 160 мм (8 дюймов)	Аксессуары	HAMMER	310.0000	3	34		шт.	5	Максидом	NULL
18	S021R4	Адаптер для шток	Адаптер для штока стеклоочистителя ALCA Top Lock	Автозапчасти	ALCA	200.0000	2	13		шт.	3	Максидом	NULL
19	S037R9	Шелка	Шелка AIRLINE AB-H-03	Аксессуары	S037R9.jpg	740.0000	2	26		шт.	25	220-volt	NULL
20	S625T5	Шелка	Шелка стеклоочистителя ALCA Start 15" 40cm карман.	Автозапчасти	ALCA	240.0000	3	12		шт.	20	Максидом	NULL
21	S626R4	Шелка	Шелка стеклоочистителя ALCA Super Flat 15" 40cm бес.	Автозапчасти	ALCA	530.0000	4	26		шт.	2	Максидом	NULL
22	S983R4	Шелка	Шелка c/o BOSCH ECO 650 650мм карманная	Автозапчасти	S983R4.jpg	500.0000	4	8		шт.	15	Максидом	NULL
23	T589T6	Термометр	Термометр ALCA 570000	Аксессуары	ALCA	1400.0000	2	3		шт.	10	220-volt	NULL
24	V892T6	Свеча зажигания	Свеча зажигания CHAMPION ESP 7FR1TC	Автозапчасти	CHAMPION	130.0000	3	21		шт.	5	Максидом	NULL
25	Z326T9	Зарядное устройство	Устройство зарядное EXPERT 3V-300 6-12В 3.0A	Зарядные устройства	EXPERT	2400.0000	3	14		шт.	15	Максидом	NULL
26	Z374R3	Зарядное устройство	Зарядное устройство AIRLINE ACH-15A-08	Зарядные устройства	Z374R3.jpg	4600.0000	2	14		шт.	25	220-volt	NULL

Рисунок 8 – Наполнитель данных