

WPF

Windows Presentation

Foundation

История развития WPF

Платформа уровня представления (UI framework) для построения графических интерфейсов

Windows Presentation Foundation (WPF) основывается на векторной системе визуализации и ориентирована на разработку клиентских Windows приложений, базирующихся на технологии Microsoft .NET.

| Выпуск | Дата | В составе | .NET Framework |
|--------|------|--|--------------------|
| 1 | 2006 | Avalon WinFX Windows Vista | 3.0 |
| 2 | 2007 | Visual Studio 2008 | 3.5 |
| 3 | 2008 | Visual Studio 2008 SP1 Windows 7 | 3.5 SP1 |
| 4 | 2010 | Visual Studio 2010 Windows след. версии | 4.0 |
| 5 | 2012 | VS2012 | 4.5 |
| 6 | 2015 | VS2015 | 4.6 |
| 7 | 2017 | VS2017 | 4.7.x .NET Core |

<https://github.com/dotnet/wpf>

Альтернативы

► Universal Windows Platform (UWP)

- C# XAML
- Веб и мобильные приложения

► Avalonia

- XAML для .NET Framework, .NET Core и Mono
- Мобильная поддержка

The screenshot displays the Microsoft Dynamics CRM Customer Model interface. At the top, there's a navigation bar with icons for Home, Accounts, Leads, Opportunities, and Contacts. Below the navigation bar, the title "Microsoft Dynamics - Your Step Business Model" and the URL "http://CRM.dynamics.com/CustomerModel/Finance" are visible.

The main area features a green header bar with the word "Finance". Underneath, there are two main sections: "Organizations" and "Processes".

Organizations: This section shows a hierarchical structure of roles:

- At the top is "Charter Manager".
- An arrow points down to "Role Manager".
- An arrow points down to "Role Controller".
- From "Role Controller", three arrows branch out to three green boxes:
 - "Case Assessment"
 - "Case Document"
 - "Profile Accounting Manager"
- From "Profile Accounting Manager", four arrows point to four green boxes:
 - "April Accounts Receivable Collector"
 - "Austin Accounts Receivable Accountant"
 - "Cassie Credit and Collections Manager"
 - "Mike Payroll Administrator"

Processes: This section lists five process groups:

- Pay
- Collect
- Treasury Management
- Capital Assets
- Close

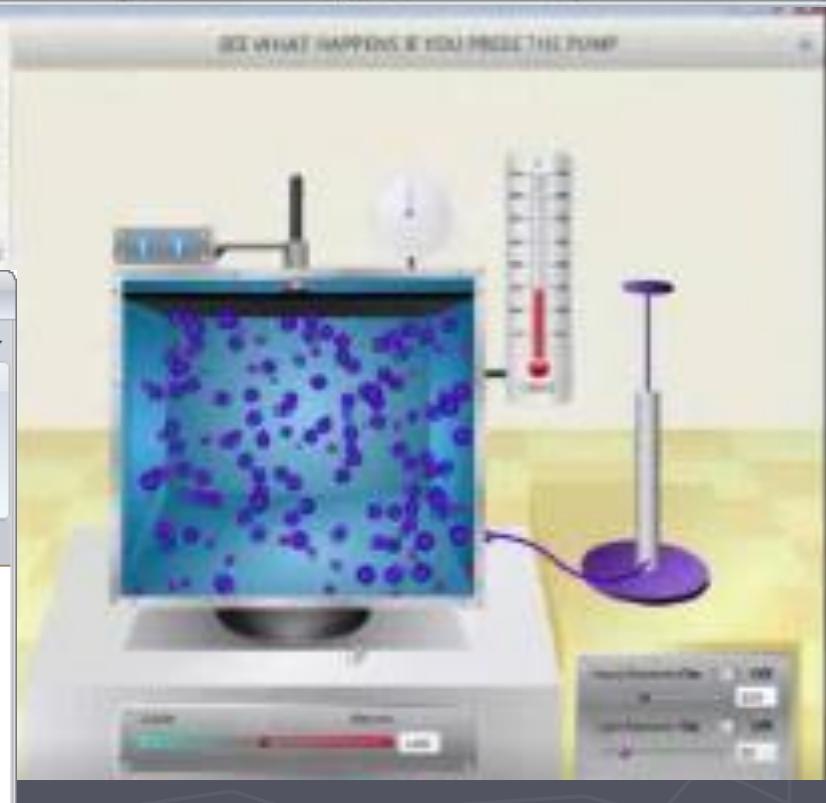
People: This section displays profiles for five individuals:

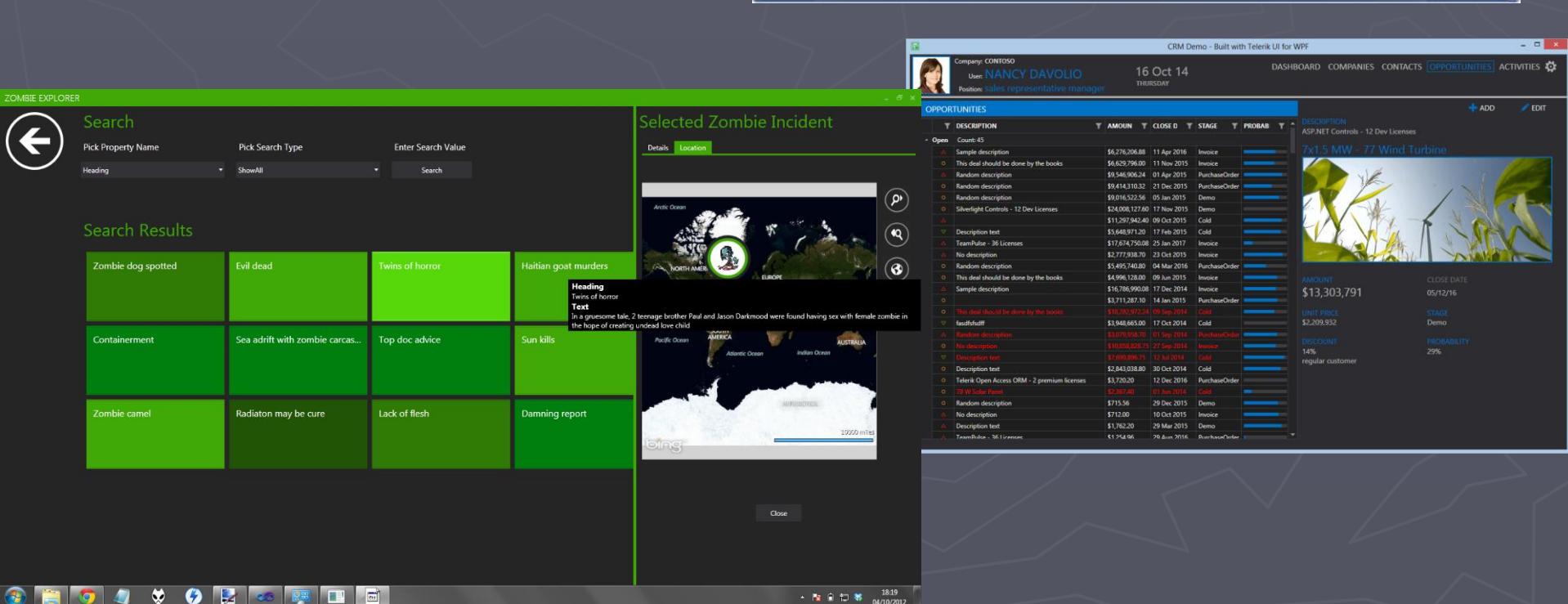
- April:** Accounts Receivable Collector. Described as handling day-to-day operations of a retail business, managing collections, paying bills, and balancing the bank statements. April handles more complex tasks of payroll, depreciation, and tax calculations.
- Austin:** Accounts Receivable Accountant. Described as a key approach-oriented and capable controller from the Project 32 assessment module to pay attention to details and ensure accuracy. Austin is also involved in legal assessments. Described as a solid financial analyst of the company's financial statements.
- Cassie:** Accounts Receivable Administrative Assistant. Described as handling payment requests and creating invoices. Cassie ensures that each transaction is appropriate and timely. She may follow up on past due accounts until they are paid.
- Mike:** Payroll Administrator. Described as a key step in the payroll process that payroll managers or audit teams can rely on for consistency. He is an administrator or employee. Mike is in charge of processing payroll within a company's financial department.
- Andy:** Credit and Collections Manager. Described as ensuring efficient collections and maintaining customer relationships. Andy handles more complex tasks of payroll, depreciation, and tax calculations.

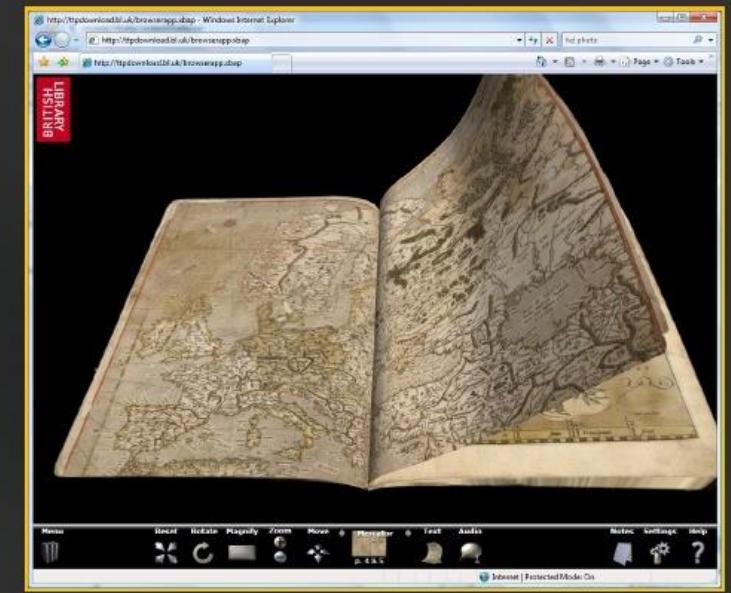
On the right side of the screen, there's a sidebar with a "Quick Edit" button and three status indicators: "Active", "In Progress", and "On Hold".

A screenshot of Microsoft Expression Blend 2. The interface shows a timeline at the top with five speech bubble icons arranged horizontally. Below the timeline is a workspace containing a single speech bubble icon. On the left, there's a tool palette with icons for selection, text, shapes, and other design tools. The bottom left shows a 'Properties' panel with tabs for 'General', 'Visual', 'Behavior', 'Interaction', 'States', and 'Visual States'. The bottom right shows a 'Timeline' panel with a timeline bar and keyframe markers. The top menu bar includes 'File', 'Edit', 'View', 'Export', 'Project', 'Debug', 'Windows', and 'Help'. The title bar says 'DesignWithMe.sln - Microsoft Expression Blend 2'.

A screenshot of the DotNetBar RibbonPad Sample application. The ribbon bar at the top has tabs for 'Tab Group' (selected), 'Write', 'Page Layout', 'Context Tab', and 'Style'. Below the ribbon, there's a toolbar with icons for Paste, Clipboard, Font, and Paragraph. The 'Font' group contains buttons for Bold (B), Italic (I), Underline (U), and Alignment (A). A color picker window titled 'Theme Colors' is open over the font group, showing a grid of standard colors. At the bottom of the color picker is a button labeled 'More Colors...' with a cursor hovering over it.







British Library

Revolutionary new 3D experience for accessing priceless literary treasures

Status: Completed

<http://www.bl.uk/onlinegallery/virtualbooks/index.html#>

BRITISH
LIBRARY

Ethiopic Bible Selections

Search

Click a page to turn or drag a page from a corner



Cover

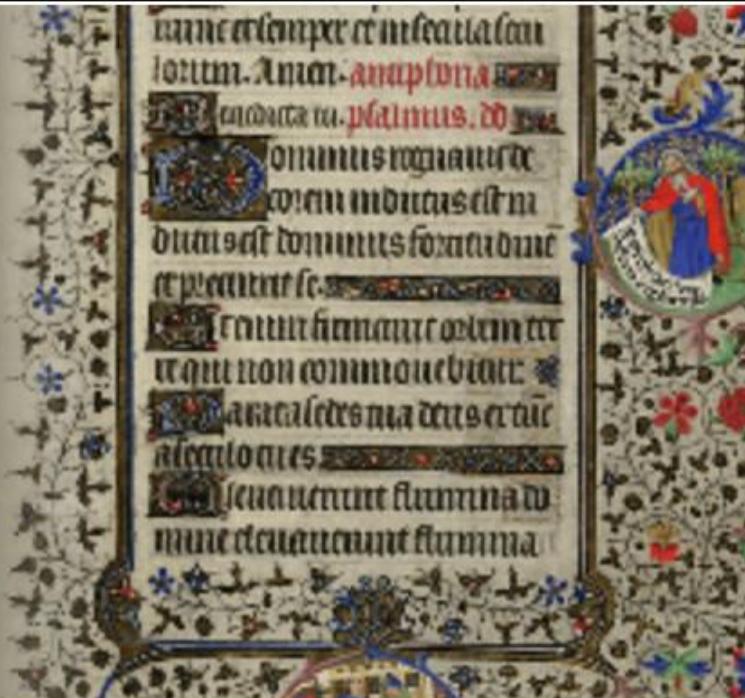
Menu | Help |

| Reset | Rotate | + Zoom | - | Move | ← → | Read | Listen |

| Comment |

The Bedford Hours

Search virtual



ff. 54-55



Menu

Help

Read

List

WinForms vs WPF

- ▶ 1) Аппаратное ускорение за счет визуализации

WinForms

User32.dll GDI/GDI+

WPF

DirectX+

аппаратная поддержка

графический процессор на видеокарте

- 2) Независимость от разрешения экрана

Незав. от устр. единиц = 1/96

DPI(dots per inch)

[Размер в физических единицах] = [Размер в независимости от устройства единицах] x [DPI системы]

Ключевые возможности

- ▶ Веб-подобная модель компоновки
- ▶ Богатая модель рисования
 - Для игр лучше спец. фрейм-ки
- ▶ Развитая текстовая модель
- ▶ Поддержка аудио и видео
- ▶ Приложения на основе страниц
- ▶ Декларативный пользовательский интерфейс (XAML)
(сочетание)
- ▶ Стили и шаблоны. Команды
 - Объем кода будет больше чем WinForms

XAML

```
<Button Width="100"> OK
<Button.Background>
LightBlue
</Button.Background>
</Button>
```

реализации
компонентов и
элементов
управления

WPF

Managed API

PresentationFramework.dll

PresentationCore.dll

WindowsBase.dll

базовые типы

Вспомогательные классы

Unmanaged API - уровень интеграции

milcore.dll

WindowsCodecs.dll

интеграция компонентов
WPF с DirectX

поддержка изображений

компоненты операционной системы

DirectX

User32

Архитектура WPF

Unmanaged

Windows Presentation Foundation

DOCUMENT SERVICES

- XPS Documents
- Packaging Services

USER INTERFACE SERVICES

- Application Services
- Deployment Services
- Controls
- Layout
- Databinding

MEDIA INTEGRATION LAYER

- Imaging
- Effects
- 2D
- 3D
- Text
- Audio
- Video
- Animation
- Composition Engine

BASE SERVICES

- XAML
- Accessibility
- Input & Eventing
- Property System

Desktop Windows Manager

Composition Engine

Windows Media Foundation

Windows Vista Display Driver (LDDM)

Print Spooler



PresentationFramework

PresentationCore

Common Language Runtime

milcore

User32

DirectX

Kernel

KELUSI

Визуальное представление



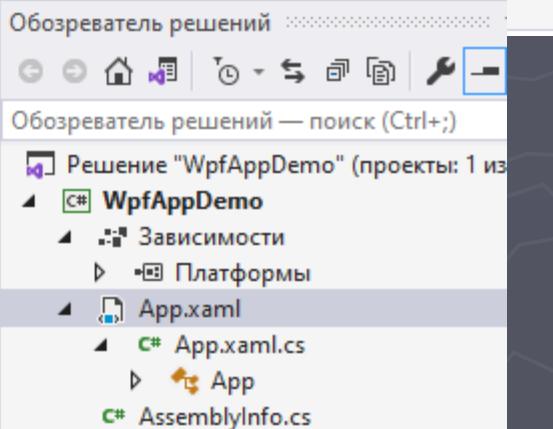
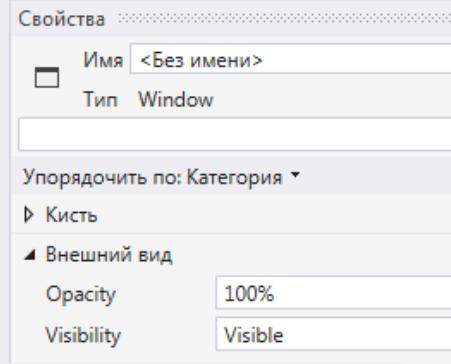
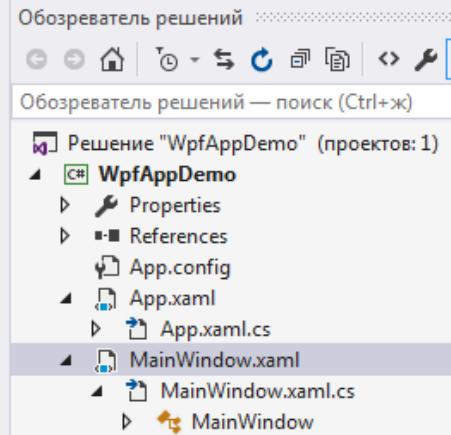
Разметка XAML

```
<Window x:Class="WpfAppDemo.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:WpfAppDemo"
    mc:Ignorable="d"
    Title="MainWindow" Height="350" Width="525">
```

94 %

Показ App.xaml App.xaml.cs MainWindow.xaml MainWindow.xaml.cs

```
1 <Application x:Class="WpfAppDemo.App"
2     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/pre-
3     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4     xmlns:local="clr-namespace:WpfAppDemo"
5     StartupUri="MainWindow.xaml">
6     <Application.Resources>
7
8         </Application.Resources>
9     </Application>
10
```



XAML: Декларативная разработка под Windows

► eXtensible Application Markup Language (XAML)

- Декларативный язык разметки для пользовательского интерфейса
- Совместная работа дизайнера и разработчика с помощью инструментов
- WCF , WF

► XAML компилируется в BAML

- BAML бинарное представление XAML оптимизированное для времени выполнения
- BAML внедряется в ресурсы сборок

Принципы XAML

- ▶ Каждый элемент в документе XAML отображается на экземпляр класса .NET.
- ▶ код XAML допускает вложение одного элемента внутрь другого
- ▶ Свойства каждого класса можно устанавливать через атрибуты или вложенные дискрипторы

XAML

```
<Button Width="100"> OK  
<Button.Background>  
LightBlue  
</Button.Background>  
</Button>
```

Стили кодирования

► *Только код (WinForms)*

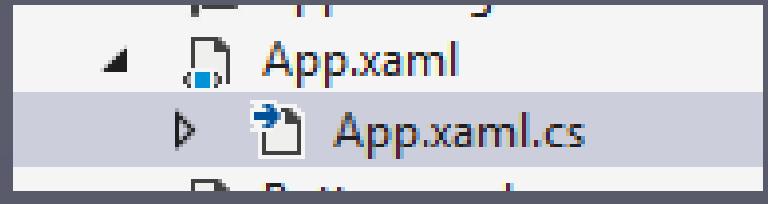
```
Button myButton = new Button();
myButton.Width = 100;
myButton.Height = 30;
myButton.Content = "Кнопка";
layoutGrid.Children.Add(myButton);
```

- *Код и не компилированная разметка (XAML)*
- *Код и компилированная разметка (BAML)*

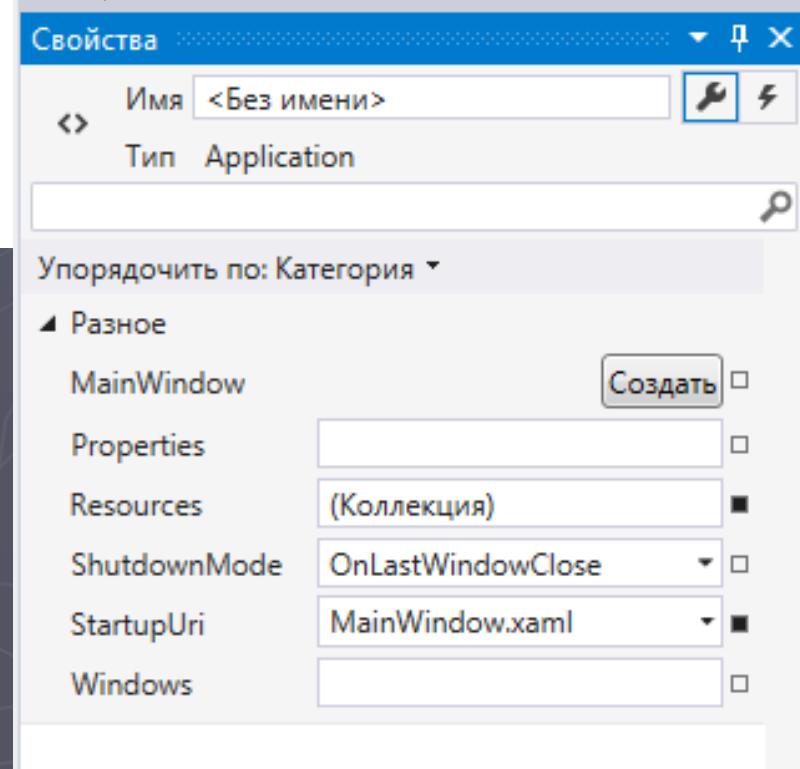
Файл App.xaml

```
<Application x:Class="WpfAppDemo.App" ← указывает на класс, который
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" будет представлять приложение
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:local="clr-namespace:WpfAppDemo"
        StartupUri="MainWindow.xaml"> ← Стартовое окно
<Application.Resources>
    ...
</Application.Resources>
</Application>
```

Ресурсы уровня
приложения –
доступны глобально
на уровне
приложения



указывает на класс, который
будет представлять приложение



Свойство **ShutdownMode**

- `OnMainWindowClose`: приложение работает, пока открыто главное окно
- `OnLastWindowClose`: приложение работает, пока открыто хотя бы одно окно
- `OnExplicitShutdown`: приложение работает, пока не будет явно вызвано `Application.Shutdown()`

События приложения

- ▶ **Startup**
- ▶ **Activated**
- ▶ **Deactivated**
- ▶ **SessionEnding**
- ▶ **DispatcherUnhandledException**
- ▶ **LoadCompleted**
- ▶ **Exit**

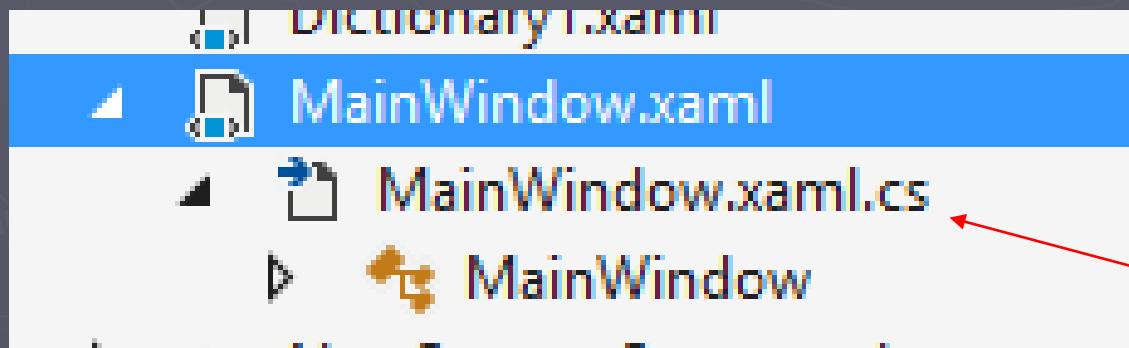
```
ССЫЛКА: 0
public partial class App : Application
{
    override
    public Li
    ссылка: 1
    public Ap
    {
        this.
        this.
    }
    public Li Equals(object obj)
    {
        * OnActivated(EventArgs e)
        * OnDeactivated(EventArgs e)
        * OnExit(ExitEventArgs e)
        * OnFragmentNavigation(System.Windows.Navigation.FragmentNavigationEventArgs e)
        * OnLoadCompleted(System.Windows.Navigation.NavigationEventArgs e)
        * OnNavigated(System.Windows.Navigation.NavigationEventArgs e)
        * OnNavigating(System.Windows.Navigation.NavigatingCancelEventArgs e)
    }
}
```

bool obj
Определ

```
public partial class App : Application
{
    protected override void OnStartup(StartupEventArgs e)
    {
        base.OnStartup(e);
        var mWin = new MainWindow();
        mWin.Show();
    }
}
```

```
public partial class MainWindow : Window  
{  
  
    public MainWindow()  
    {  
  
        InitializeComponent();  
        App.Current.
```

Можно обращаться в
приложению



логика приложения

► obj/Debug App.g.cs

```
namespace WpfAppDemo {
public partial class App : System.Windows.Application {

    [System.Diagnostics.DebuggerNonUserCodeAttribute()]
    [System.CodeDom.Compiler.GeneratedCodeAttribute("PresentationBuildTasks", "4.0.0.0")]
    public void InitializeComponent() {

        #line 5 "..\..\App.xaml"
        this.StartupUri = new System.Uri("MainWindow.xaml", System.UriKind.Relative);

        #line default
        #line hidden
    }

    [System.STAThreadAttribute()]
    [System.Diagnostics.DebuggerNonUserCodeAttribute()]
    [System.CodeDom.Compiler.GeneratedCodeAttribute("PresentationBuildTasks", "4.0.0.0")]
    public static void Main() {
        WpfAppDemo.App app = new WpfAppDemo.App();
        app.InitializeComponent();
        app.Run();
    }
}
```

WPF требует, чтобы главный поток работал однопоточном подразделении (Single-thread apartment),

Настройки окна

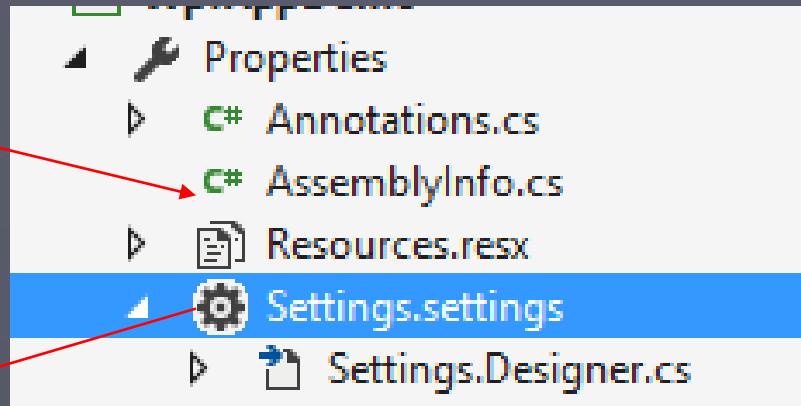
Атрибуты настройки сборки

| Имя | Тип | Область. | Значение |
|-------|--------|--------------|---|
| Title | string | Приложение | WpfDemoWindow |
| Data | string | Пользователь | bool byte char decimal double float int long sbyte short string |

Вывод

Показать выходные данные

```
"WpfAppDemo.exe" (CLR) System.Collections.Specialized.StringCollection  
"WpfAppDemo.exe" (CLR) System.DateTime  
"WpfAppDemo.exe" (CLR) System.Drawing.Color  
"WpfAppDemo.exe" (CLR) System.Drawing.Font  
"WpfAppDemo.exe" (CLR) System.Drawing.Point  
"WpfAppDemo.exe" (CLR) System.Drawing.Size  
"WpfAppDemo.exe" (CLR) System.Guid  
"WpfAppDemo.exe" (CLR) System.TimeSpan
```



Приложение – изменять не
можем
Пользователь – можно
изменять значения

Properties.Settings.Default.Save();

```
String TitleWind= Properties.Settings.Default.Title ;  
Properties.Settings.Default.Data = new System.Drawing.Point (10,20);
```

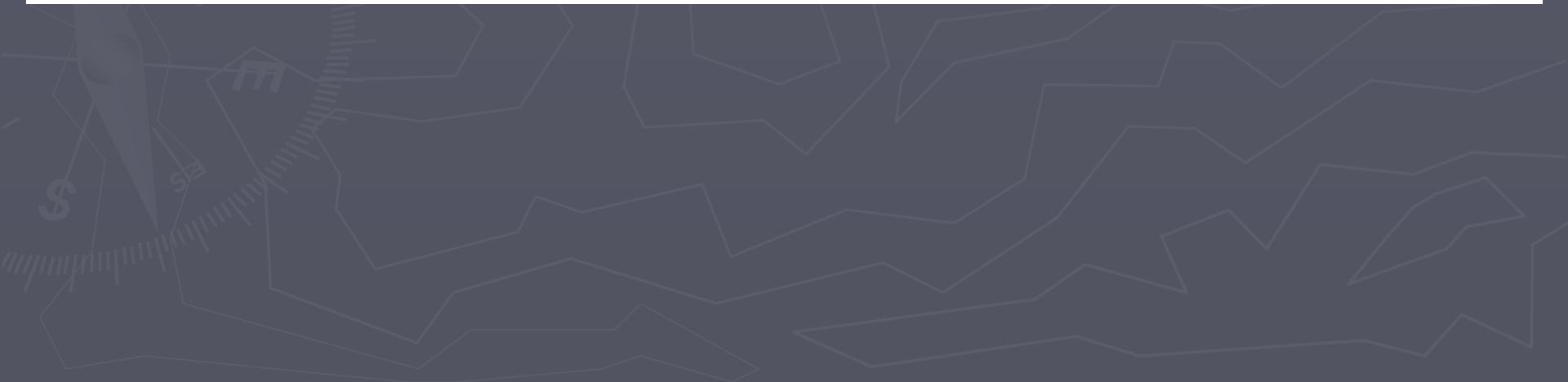
(Строка подключения)
Обзор...

Список ошибок

Вывод

```
[global::System.Configuration.ApplicationScopedSettingAttribute()]
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.Configuration.DefaultSettingValueAttribute("WpfDemoWindow")]
public string Title {
    get {
        return ((string)(this["Title"]));
    }
}

[global::System.Configuration.UserScopedSettingAttribute()]
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.Configuration.DefaultSettingValueAttribute("0, 0")]
public global::System.Drawing.Point Data {
    get {
        return ((global::System.Drawing.Point)(this["Data"]));
    }
    set {
        this["Data"] = value;
    }
}
```



XAML

Пространства имен

Описание и определение ЭУ

```
<Window x:Class="WpfAppDemo.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:WpfAppDemo"
```

определяет свойства

```
Title="MainWindow" Height="350" Width="525">
```

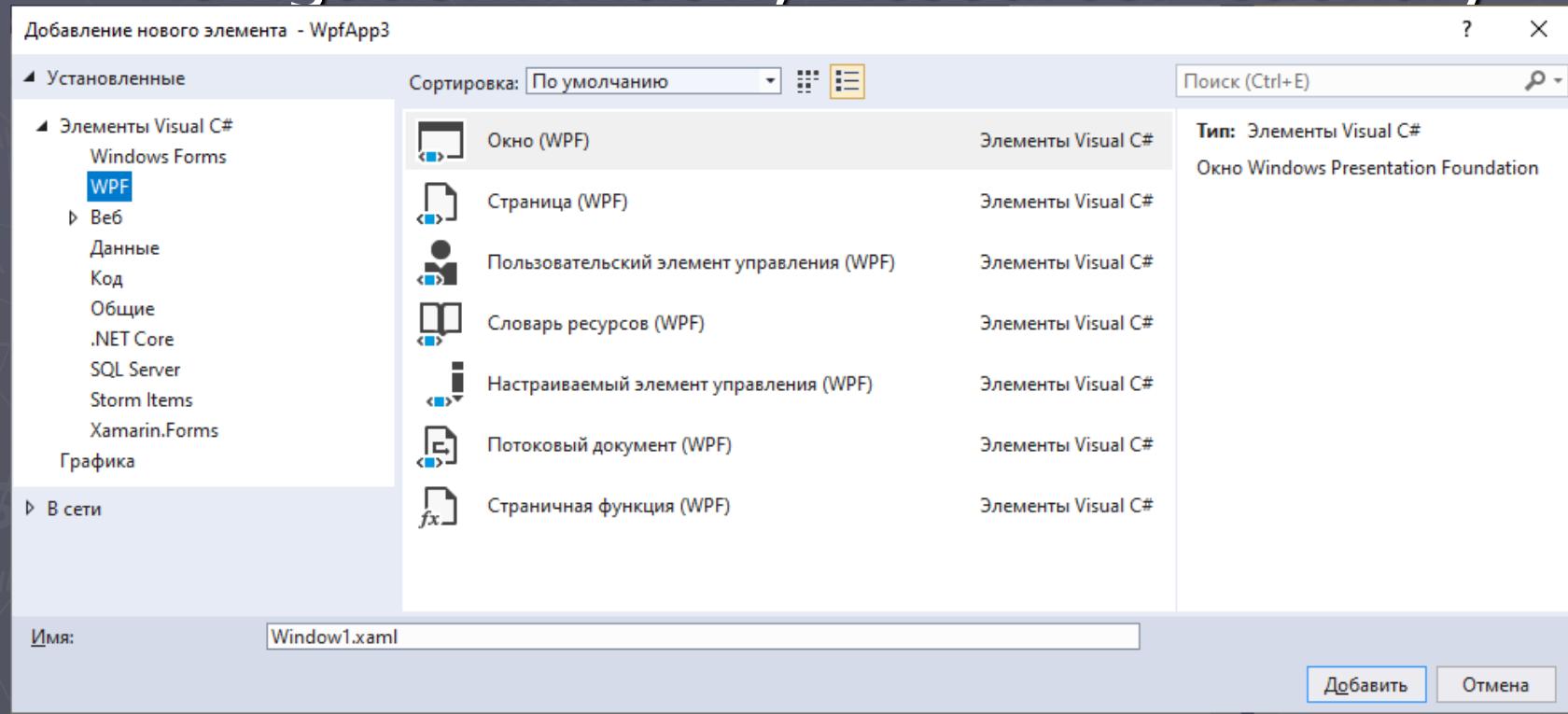
обеспечивает режим совместимости
разметок XAML

пространство имен текущего прое

```
xmlns:col ="clr-namespace:System.Collections;assembly=System"
```

Элементы XAML верхнего уровня

- ▶ Window
- ▶ Page
- ▶ Application
- ▶ Navigationwindow, ResourceDictionary



```
<StackPanel Grid.Row="2" Grid.Column="2" >
    <Button x:Name ="ButtComand"
        Click ="ButtComand_OnClick"
        FontSize="20"
        Content="Вопрос 1" />

    <Button Name ="ButtRes"
        FontSize="20"
        Click="ButtRes_OnClick"
        Content="Вопрос2" />

    <Button><!-- "Вопрос 3" -->
    <Button/>
</StackPanel>
```

XAML по умолчанию
убирает все пробелы

`xml:space="preserve"`

&
"
< и >

| Символ | Код |
|--------|--------|
| < | < |
| > | > |
| & | & |
| " | " |

► Свойства (TypeConverter)

► Сложные свойства

РодительскийЭлемент.ИмяСвойства

```
<Button x:Name="Button1" Height="30" Width="100"
        Content="Ok"
        Click="Button1_OnClick" Margin="208,126,209,164">
    <Button.HorizontalAlignment>
        Center
    </Button.HorizontalAlignment>
    <Button.Background>
        <SolidColorBrush Opacity="0.1" Color="Red" />
    </Button.Background>

</Button>
```

► Расширения разметки XAML

Элемент УстанавливаемоСвойство = "{РасширениеРазметки}"/>

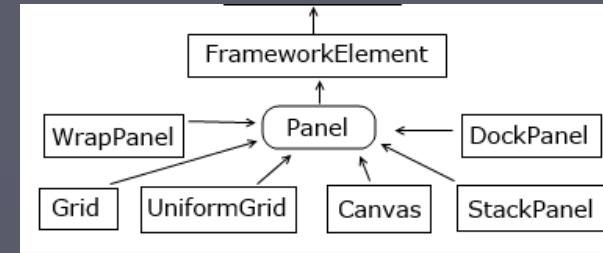
```
<Label Content="{x:Static CorLib:Environment.OSVersion}"></Label>
```

Attached property

```
<Button x:Name="Button1" Height="100" Width="130"
        Grid.Column="1" Grid.Row="1"
        Click="Button1_OnClick" >
    <Button.Content>
        OK
    </Button.Content>
    <Button.HorizontalAlignment>
        Center
    </Button.HorizontalAlignment>
    <Button.Background>
        <!--<SolidColorBrush Opacity="0.1" Color="Red" /><!--
        &lt;LinearGradientBrush&gt;
            &lt;GradientStop Color="Chocolate"
                Offset="0"&gt;&lt;/GradientStop&gt;
            &lt;GradientStop Color="BlueViolet"
                Offset="1"&gt;&lt;/GradientStop&gt;
        &lt;/LinearGradientBrush&gt;
    &lt;/Button.Background&gt;
&lt;/Button&gt;</pre>
```

Контейнеры компоновки

процесс размещения элементов
внутри контейнера



➤ Layouts

➤ Canvas

➤ UniformGrid

➤ Grid

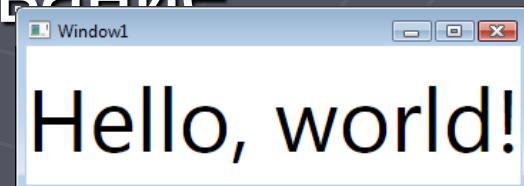
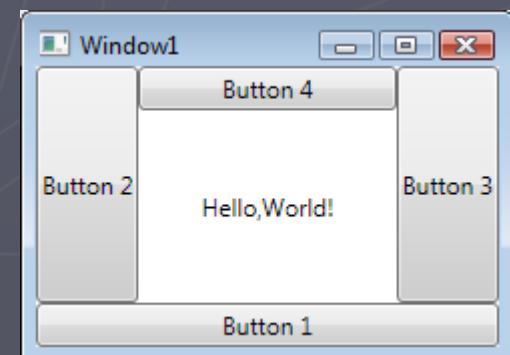
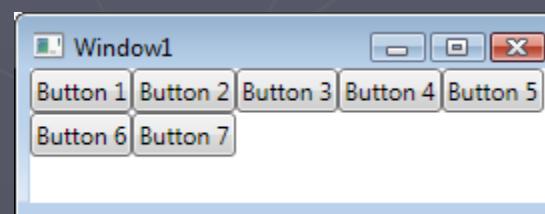
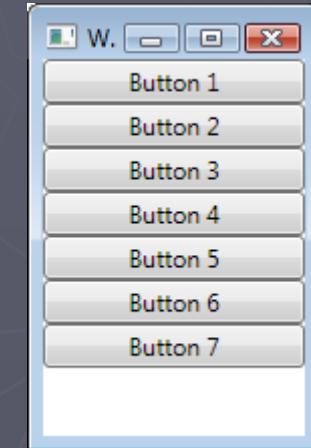
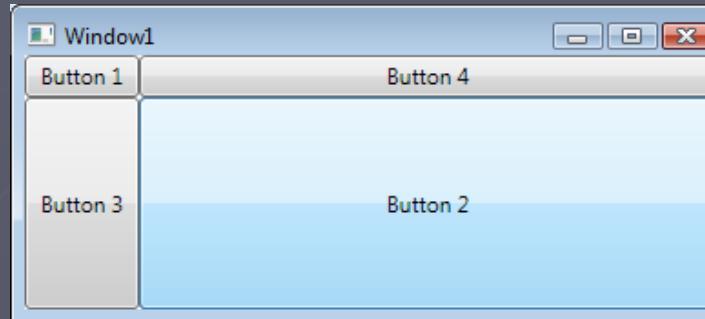
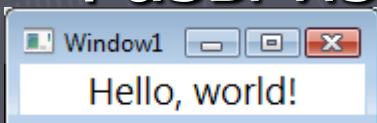
➤ StackPanel

➤ WrapPanel

➤ DockPanel

➤ ViewBox (масштабирование)

● Фазы измерение и упорядочивание



Grid

Column = 0

Column = 1

Column = 2

Row = 0

Row = 1

Row = 2

Row = 3

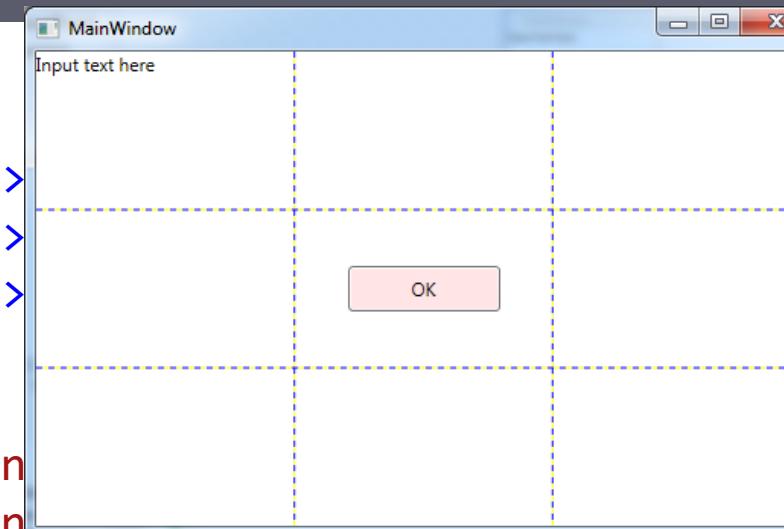


```

<Grid ShowGridLines="True">
    <Grid.RowDefinitions>
        <RowDefinition></RowDefinition>
        <RowDefinition></RowDefinition>
        <RowDefinition></RowDefinition>
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition></ColumnDefinition>
        <ColumnDefinition></ColumnDefinition>
        <ColumnDefinition></ColumnDefinition>
    </Grid.ColumnDefinitions>
    <Button x:Name="Button1" Height="30" Width="100"
            Grid.Column="1" Grid.Row="1"
            Click="Button1_OnClick" >
        // ...
    </Button>
    <TextBlock Name="TextBlock1" Text="Input text here"
            Grid.Column="0"
            Grid.Row="0"/>
</Grid>

```

**Абсолютные , Автоматические
Пропорциональные размеры**



```

<ColumnDefinition Width="150" />
<RowDefinition Height="10 px" />
<ColumnDefinition Width="*" />
<ColumnDefinition Width="0.5*" />
<ColumnDefinition Width="1.5*" />
<ColumnDefinition Width="Auto" />

```

UniformGrid

- Аналогичен контейнеру Grid → все столбцы и строки одинакового размера

```
<UniformGrid Rows="2" Columns="2">
```

```
...
```

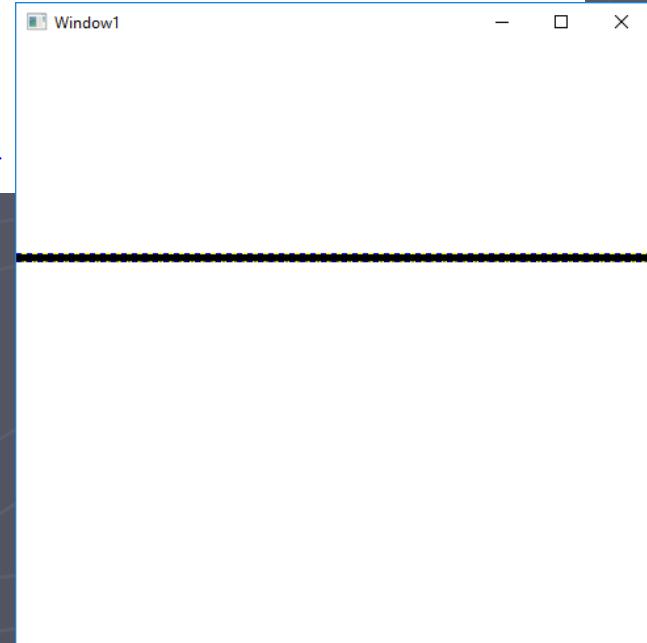
GridSplitter

разделитель между столбцами или строками

Сдвиг - регулирует ширину столбцов и высоту строк

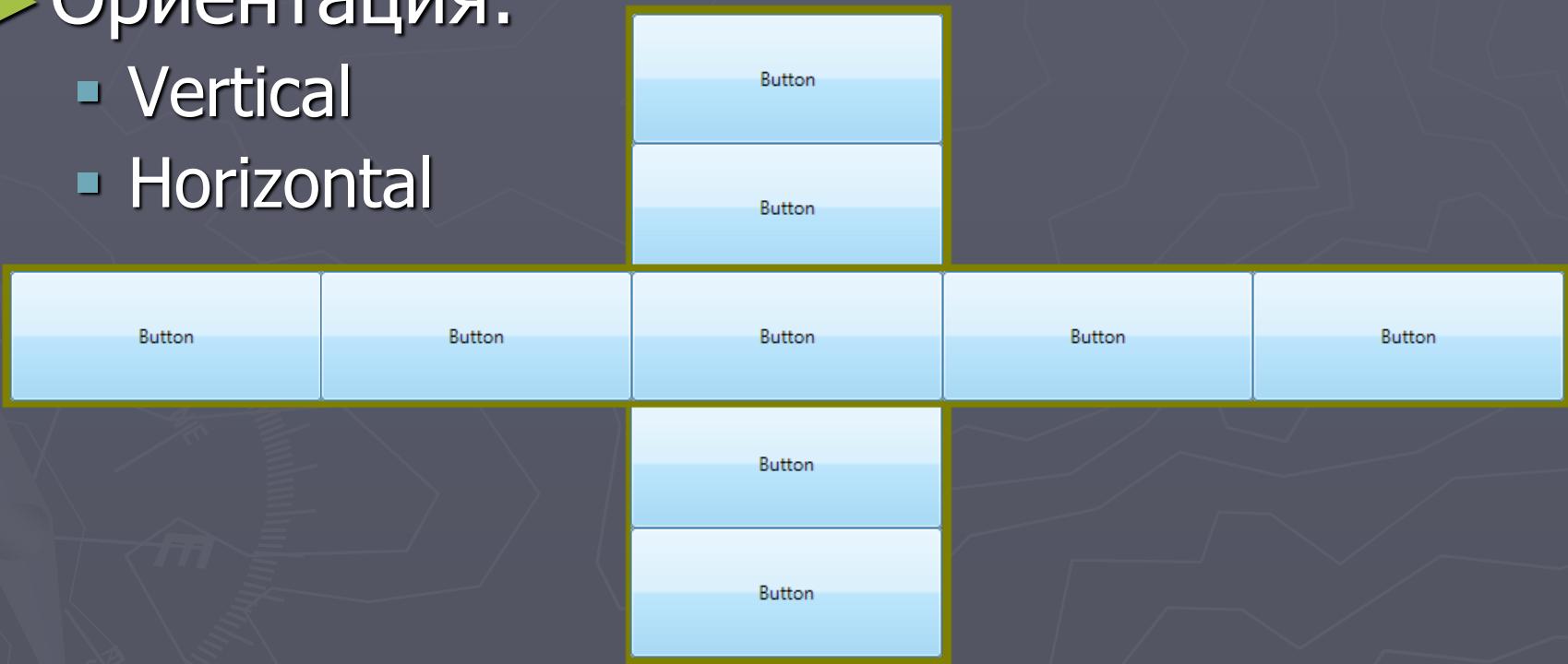
```
<Grid ShowGridLines="True">
    <Grid.RowDefinitions>
        <RowDefinition Height="*"/>
        <RowDefinition Height="Auto"/>
        <RowDefinition Height="*"/>
    </Grid.RowDefinitions>

    <GridSplitter Grid.Row = " 1"
        ShowsPreview="false"
        Height="6"
        Background="Black"
        HorizontalAlignment="Stretch"/>
```

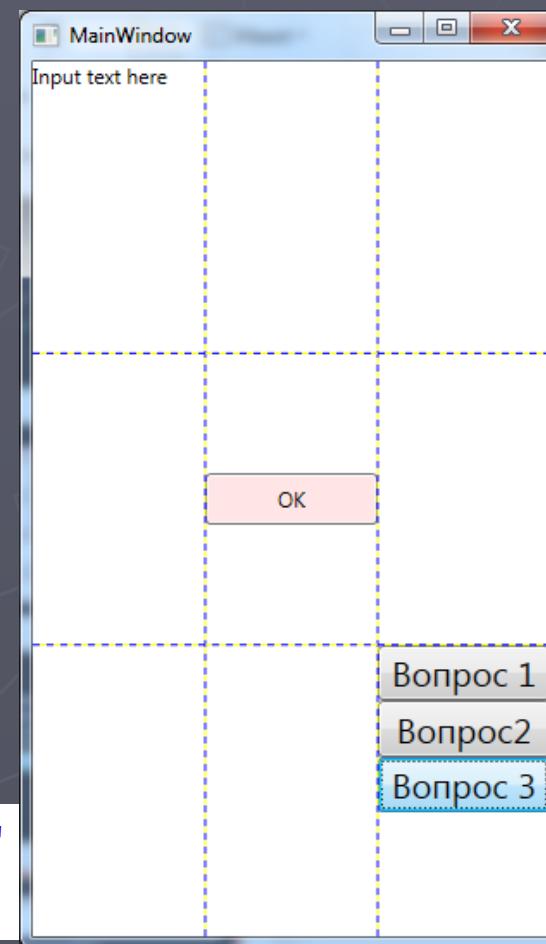
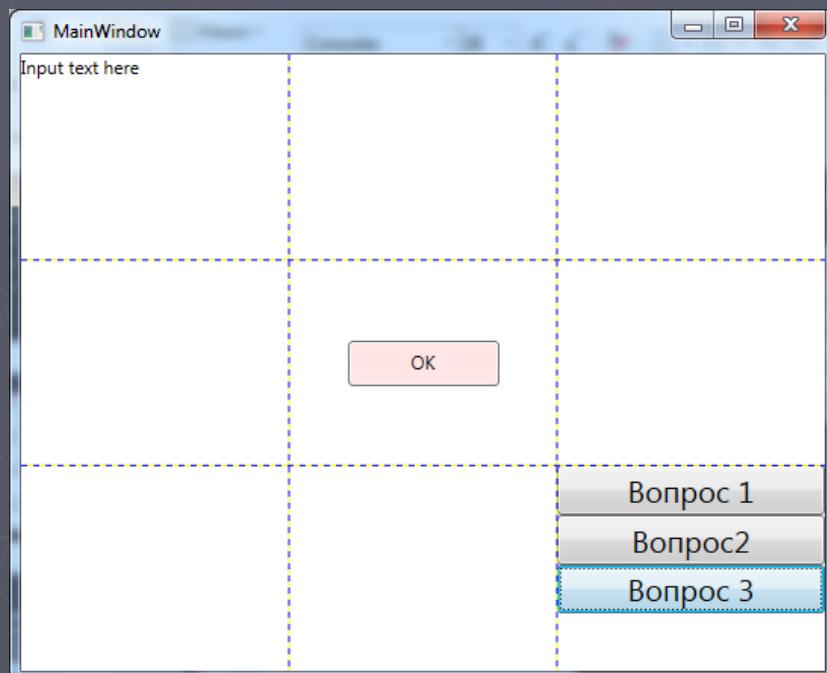


StackPanel

- ▶ Ставит в ряд дочерние элементы
- ▶ Ориентация:
 - Vertical
 - Horizontal

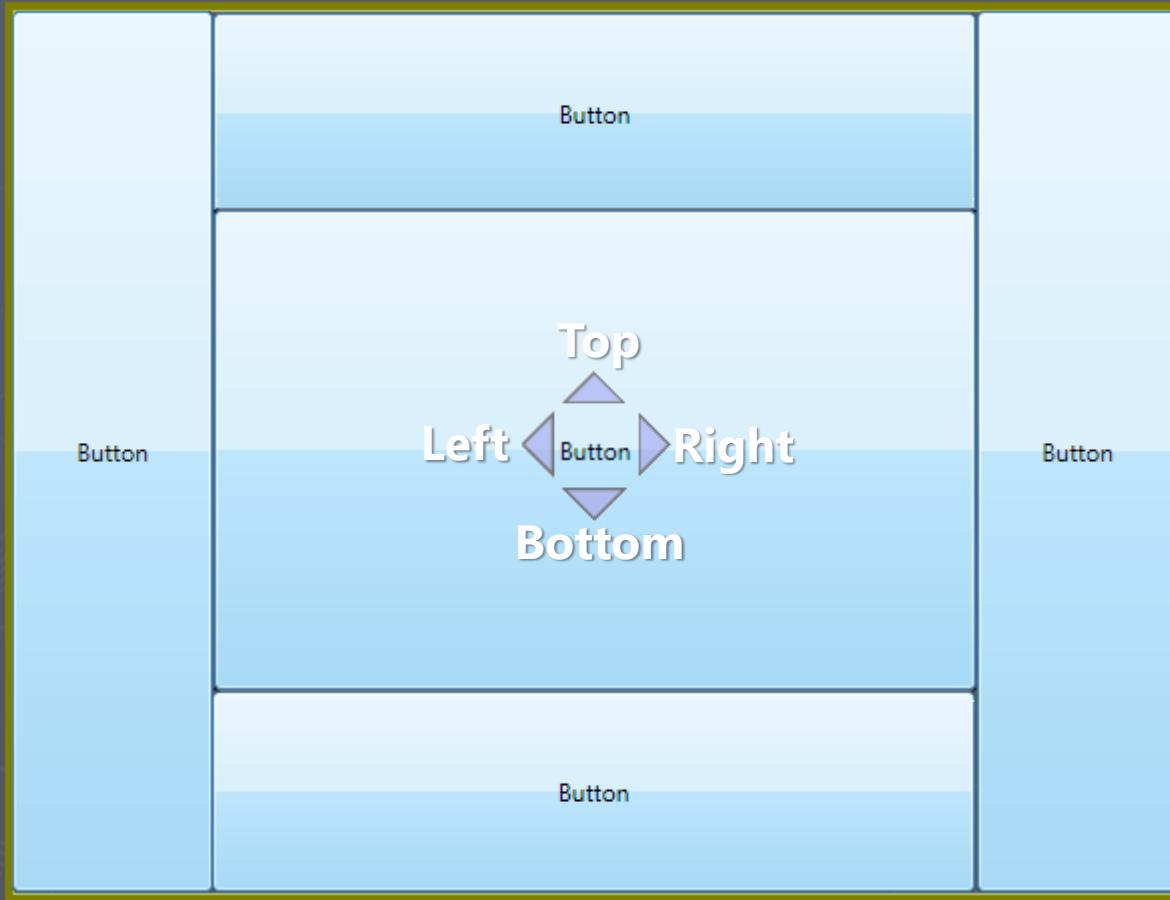


```
<StackPanel Grid.Row="3" Grid.Column="3" >
    <Button FontSize="20" Content="Вопрос 1" />
    <Button FontSize="20" Content="Вопрос2" />
    <Button FontSize="20" Content="Вопрос 3" />
</StackPanel>
```



```
<StackPanel Orientation="Horizontal"
FlowDirection="RightToLeft">
```

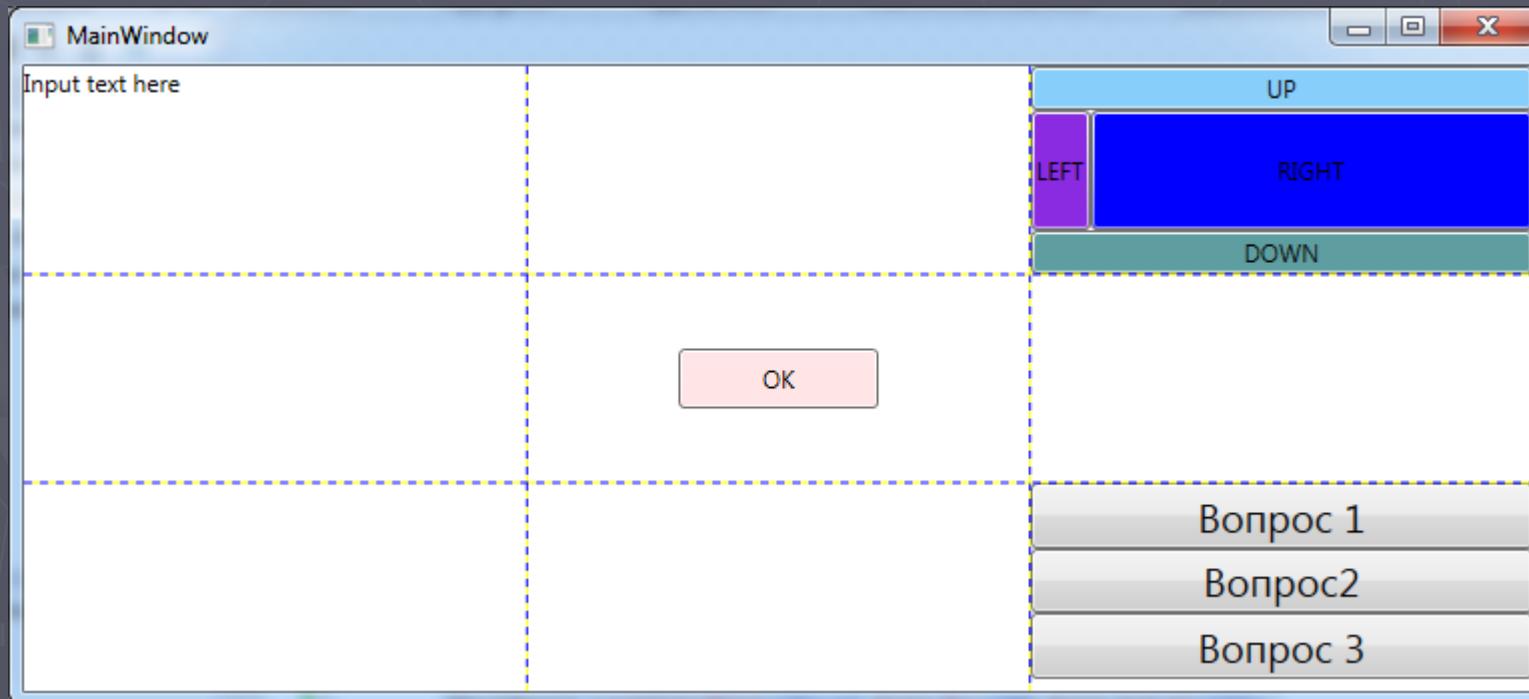
DockPanel



LastChildFill

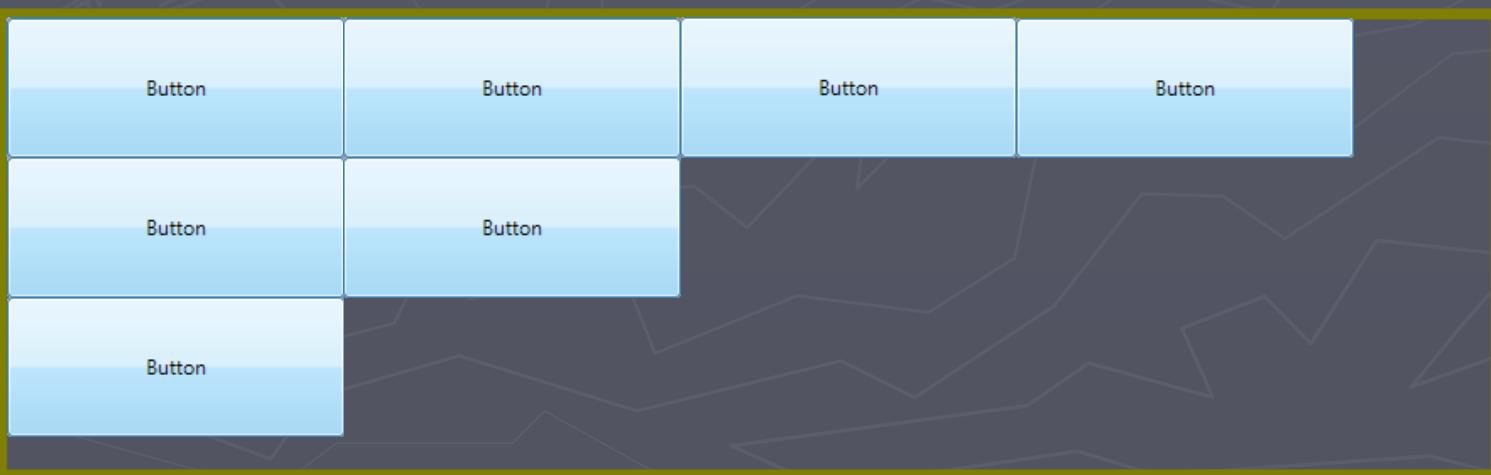
```
<DockPanel LastChildFill="True" Grid.Row="0" Grid.Column="3">
    <Button DockPanel.Dock="Top"
        Background="LightSkyBlue" Content="UP" />
    <Button DockPanel.Dock="Bottom"
        Background="CadetBlue" Content="DOWN" />
    <Button DockPanel.Dock="Left"
        Background="BlueViolet" Content="LEFT" />
    <Button DockPanel.Dock="Right"
        Background="Blue" Content="RIGHT" />
</DockPanel>
```

LastChildFill="True",

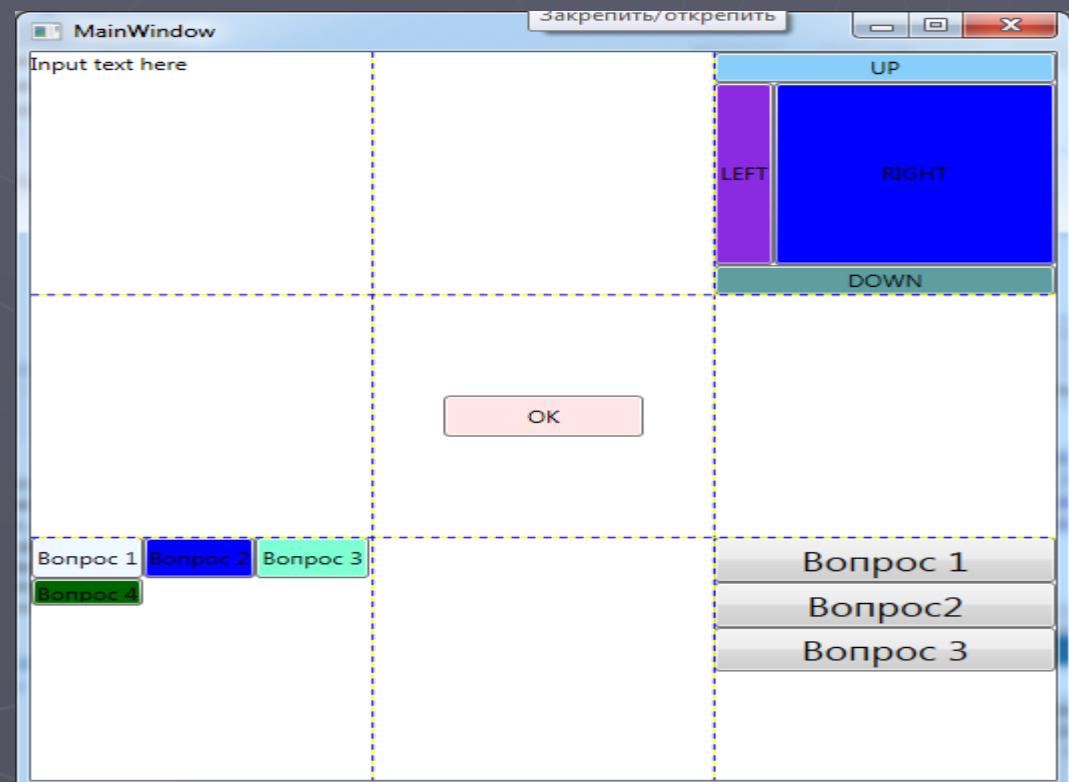
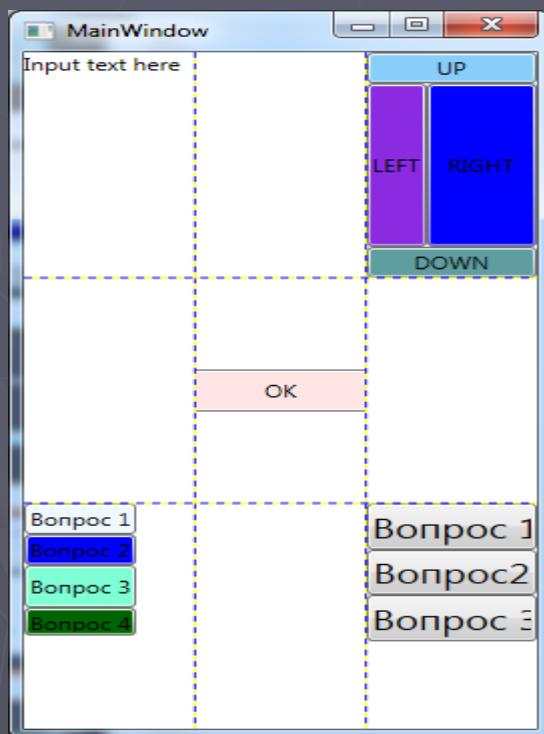


WrapPanel

- ▶ Ставит элементы в ряд
- ▶ Когда элементы не помещаются, они переносятся
- ▶ Ориентация:
 - Horizontal
 - Vertical



```
<WrapPanel Grid.Row="3">
    <Button Background="AliceBlue" Content="Вопрос 1" />
    <Button Background="Blue" Content="Вопрос 2" />
    <Button Background="Aquamarine"
        Content="Вопрос 3" Height="30"/>
    <Button Background="DarkGreen"
        Content="Вопрос 4" Height="20"/>
</WrapPanel>
```

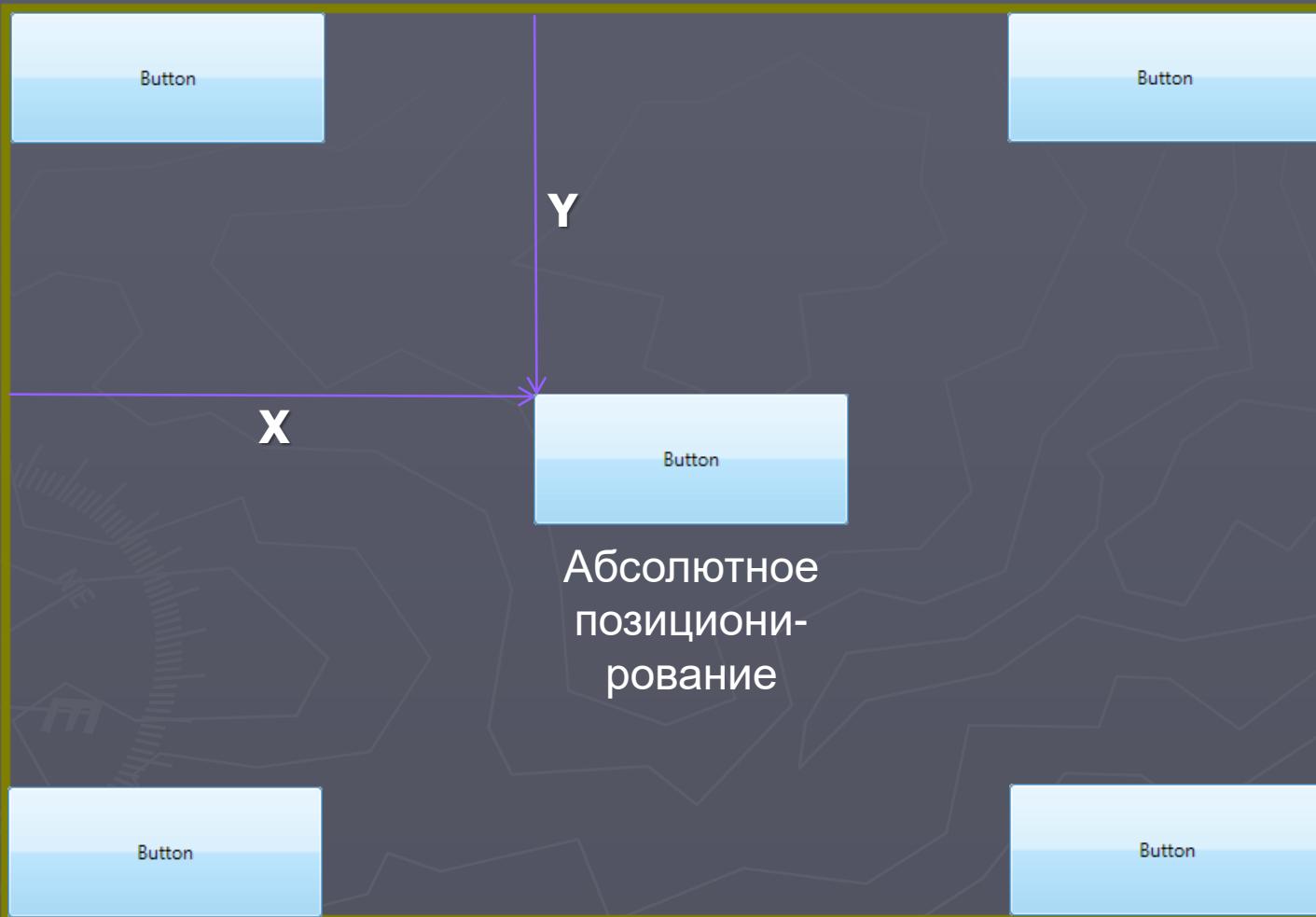


```
<WrapPanel ItemHeight="30" ItemWidth="80"
Orientation="Horizontal">
```

Canvas

Top, Left

Top, Right



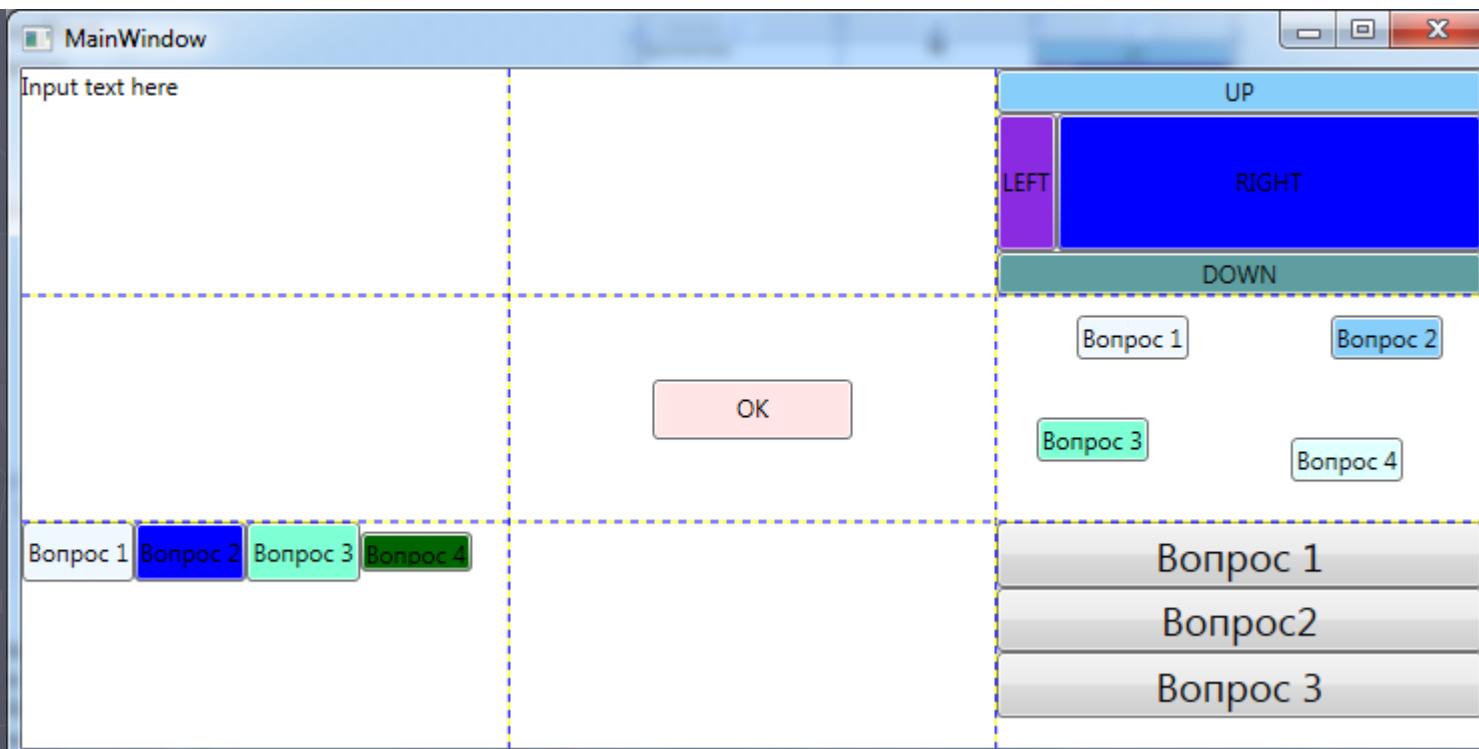
Абсолютное
позициони-
рование

Z-index – позволяет
создавать слои

Bottom,
Left

Bottom,
Right

```
<Canvas Grid.Row="1" Grid.Column="2">
    <Button Background="AliceBlue" Content="Вопрос 1"
        Canvas.Top="10" Canvas.Left="40" />
    <Button Background="LightSkyBlue" Content="Вопрос 2"
        Canvas.Top="10" Canvas.Right="20"/>
    <Button Background="Aquamarine" Content="Вопрос 3"
        Canvas.Bottom="30" Canvas.Left="20"/>
    <Button Background="LightCyan" Content="Вопрос 4"
        Canvas.Bottom="20" Canvas.Right="40"/>
</Canvas>
```



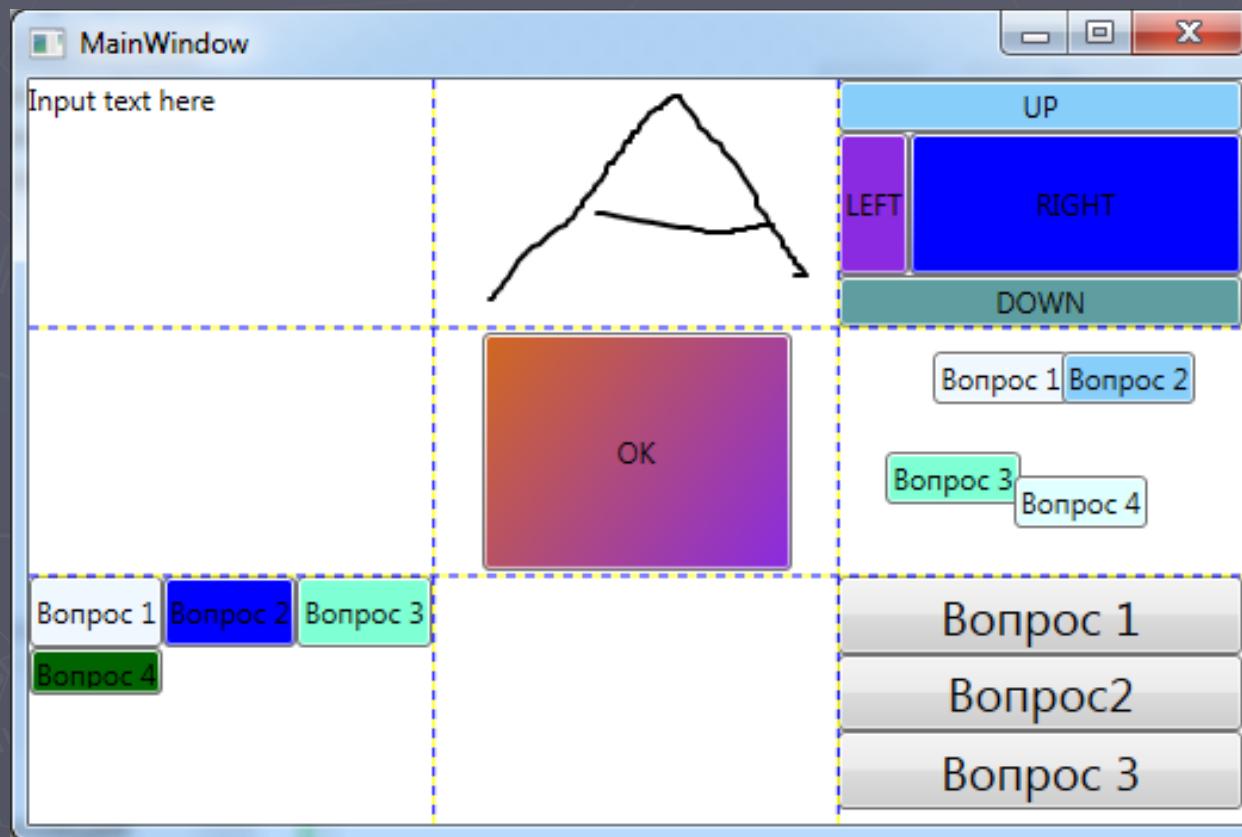
InkCanvas

InkCanvasEditMode

- обеспечение первого ввода, считывание жестов пользователя

| Имя | Описание |
|---------------|--|
| Ink | InkCanvas позволяет пользователю рисовать аннотации. Когда пользователь рисует мышью или пером, появляются штрихи. |
| GestureOnly | В этом режиме InkCanvas не позволяет пользователю рисовать аннотации, но считывает жесты, произведенные пользователем с помощью мыши или пера. |
| InkAndGesture | InkCanvas позволяет пользователю рисовать штриховые аннотации и также распознает предопределенные жесты. |
| EraseByStroke | InkCanvas удаляет весь штрих при щелчке. |
| EraseByPoint | InkCanvas удаляет часть штриха (точку штриха) при щелчке на соответствующей его части. |
| Select | InkCanvas позволяет пользователю выбирать элементы, хранящиеся в коллекции Children. Как только элемент выбран, его можно перемещать, изменять размер или удалять. |
| None | InkCanvas игнорирует ввод с помощью мыши или пера. |

```
<InkCanvas Grid.Row="0" Grid.Column="1" EditingMode = "Ink">  
    </InkCanvas>
```



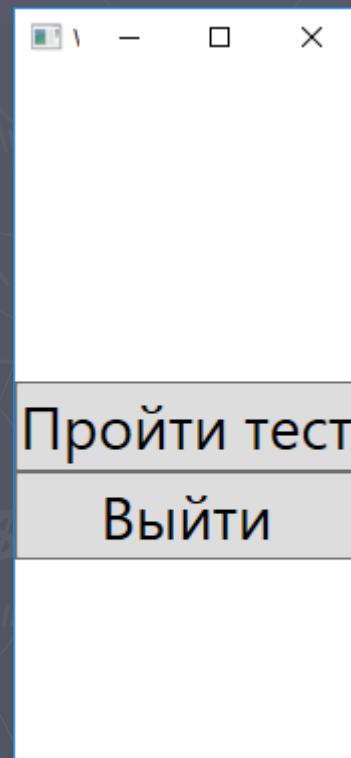
Viewbox

- ▶ Элемент (графика) может самостоятельно подгонять свои размеры к окну

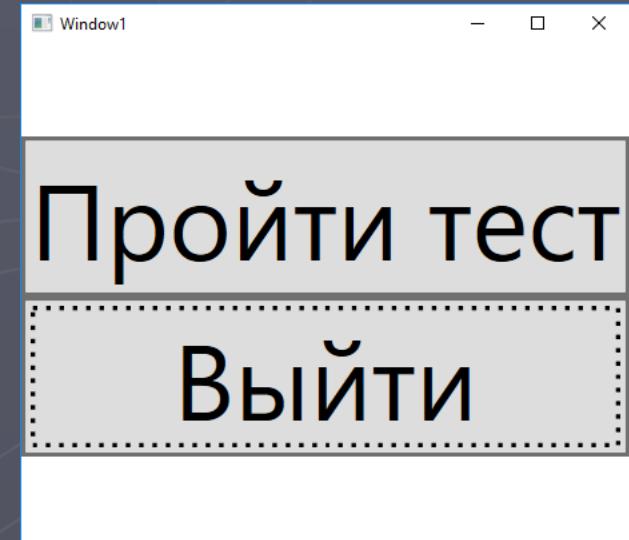
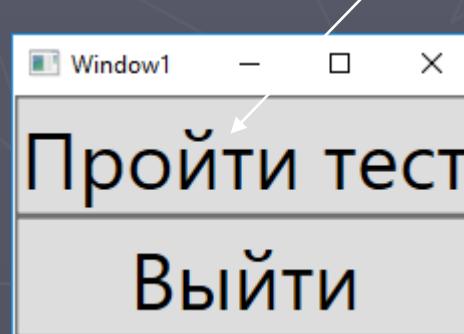
Свойства

- 1) Наследник Decorator
- 2) принимает единственный дочерний элемент (контейнер компоновки)
- 3) синхронно изменяет размеры дочерних элементов (векторная графика)

```
<Grid>
    <Viewbox>
        <StackPanel>
            <Button Name="test"
                Content="Пройти тест"/>
            <Button Name="goout"
                Content="Выйти"/>
        </StackPanel>
    </Viewbox>
</Grid>
```



увеличивает значение DPI.



Свойства для компоновки

| Имя |
|----------------------|
| HorizontalAlignment |
| VerticalAlignment |
| Margin |
| MinWidth и MinHeight |
| MaxWidth и MaxHeight |
| Width и Height |

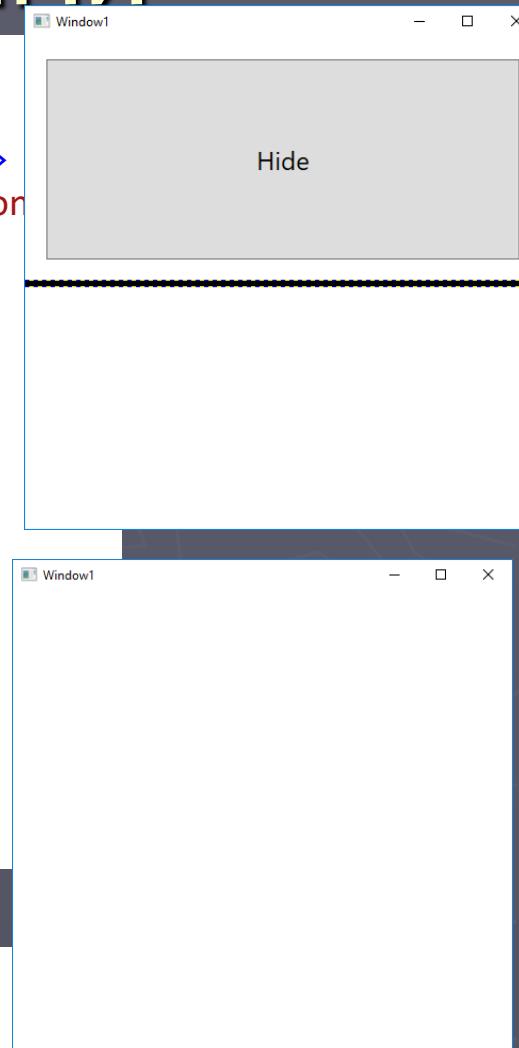
Left, Right, Center,
Stretch

Управление панелями

```
<Grid Name ="GridEx" ShowGridLines="True">
    <Grid.RowDefinitions>
        <RowDefinition Height="*"/>
        <RowDefinition Height="Auto"/>
        <RowDefinition Height="*"/>
    </Grid.RowDefinitions>

    <GridSplitter Name="GSplitter" Grid.Row =" 1"
        ShowsPreview="false"
        Height="6"
        Background="Black"
        HorizontalAlignment="Stretch"/>
    <Button Name="Ok"
        Grid.Row ="0" Content="Hide"
        Click="Ok_Click"
        Margin="20"/>

</Grid>
```



```
private void Ok_Click(object sender, RoutedEventArgs e)
{
    this.GridEx.Visibility = Visibility.Hidden;
}
```



Структура документа

[Window]

[Grid]

- Button1
- TextBlock1
- [StackPanel]
 - ButtComand
 - ButtRes
 - [Button] "Вопрос 3"
- [DockPanel]
 - [Button] "UP"
 - [Button] "DOWN"
 - [Button] "LEFT"
 - [Button] "RIGHT"
- [WrapPanel]
 - [Button] "Вопрос 1"
 - Button2
 - Button3
 - ButtonT
- [InkCanvas]
- [Canvas]
 - [Button] "Вопрос 1"
 - [Button] "Вопрос 2"
 - [Button] "Вопрос 3"
 - [Button] "Вопрос 4"
- [Path]

App.xaml.cs*

MainWindow.xaml*

Student.cs

App.config

OK

Вопрос 1 Вопрос 2

Вопрос 3 Вопрос 4

Вопрос 1
Вопрос 2
Вопрос 3

Дерево документа

```
1="0"  
3"/>  
  
'2" Grid.Column="2" >  
:tComand" Click ="ButtComand_OnClick" FontSize="20" Content="Вопрос 1" />  
:ttRes" FontSize="20" Click="ButtRes_OnClick" Content="Вопрос2" />  
'20" Content="Вопрос 3" />
```

Готово

Объектная модель WPF

DispatcherObject

Dependency Object

ContentElement

Visual

Visual3D

Freezable

FrameworkContentElement

UIElement

Animatable

FrameworkElement

Decorator

Control

Panel

Shape



Visual

► Обеспечивает:

- Рендеринг
- Clipping (Обрезание краев)
- Трансформации
- Вычисление границ

Используется для
визуализации и связи с
milcore



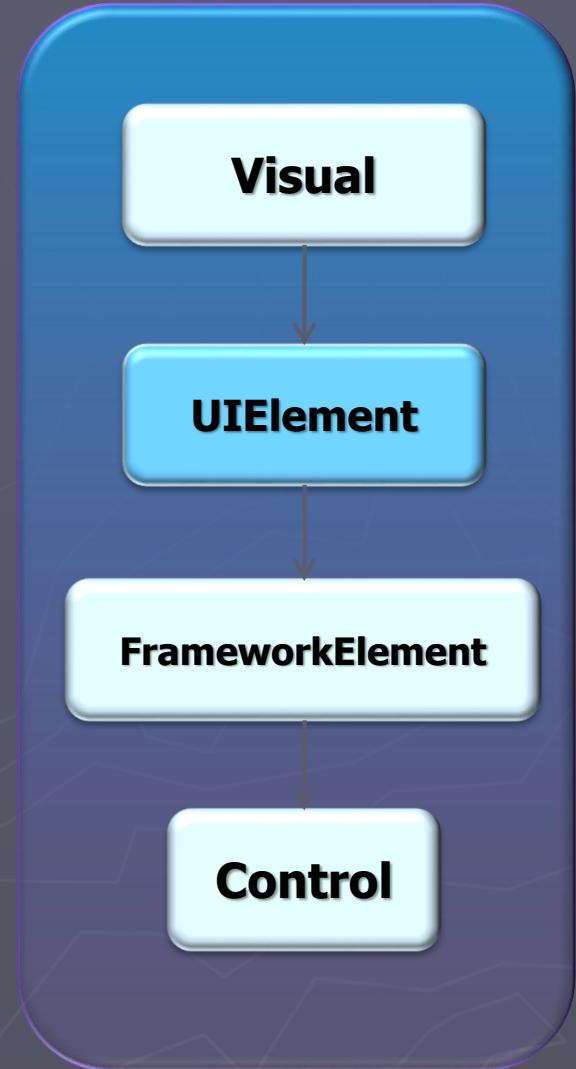
UIElement

► Добавляется:

- компоновка (*layout*),
- ввод (*input*),
- фокус (*focus*)
- события (*events*)

LIFE

процесс измерения и организации
компоновки



FrameworkElement

► Обеспечивает:

- Систему метаданных свойств
- Дополнительные свойства расположения
- Стили
- Storyboards
- Триггеры

Добавляет события и методы



Control

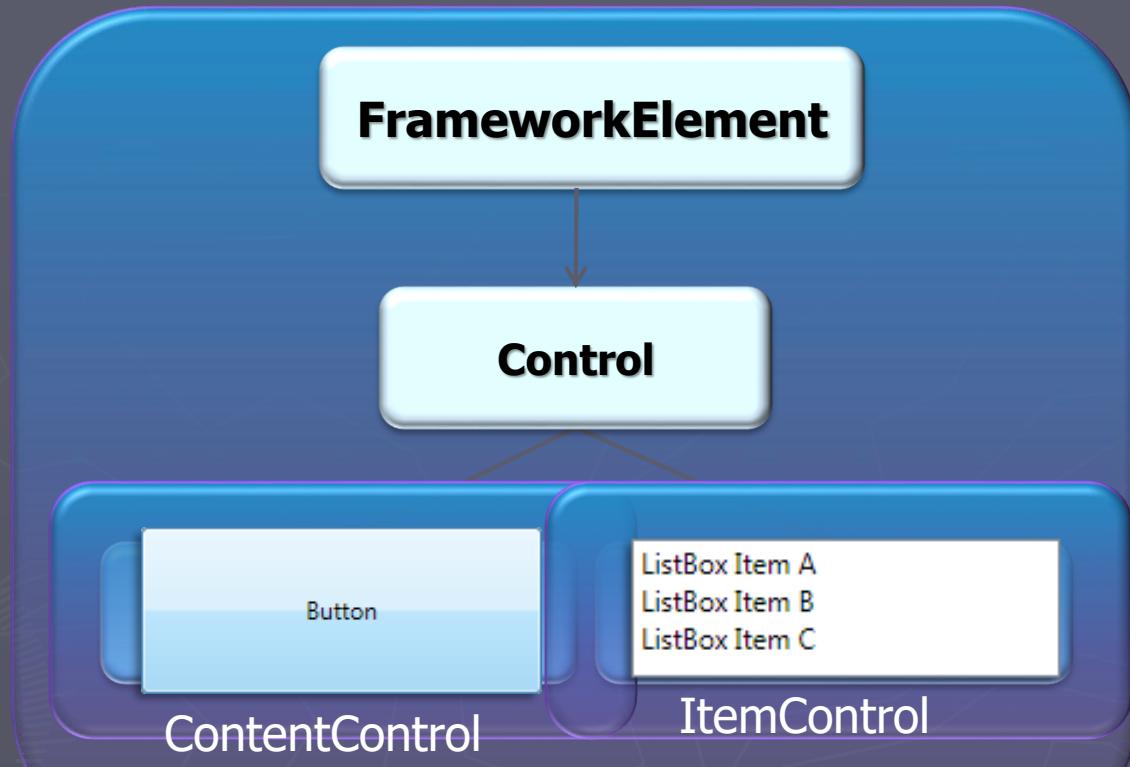
► Обеспечивает:

- дополнительные свойства для установки шрифта
- цветов переднего плана и фона

поддержка шаблонов
размеры элемента управления
прозрачность,
порядок обхода



Базовые классы Control

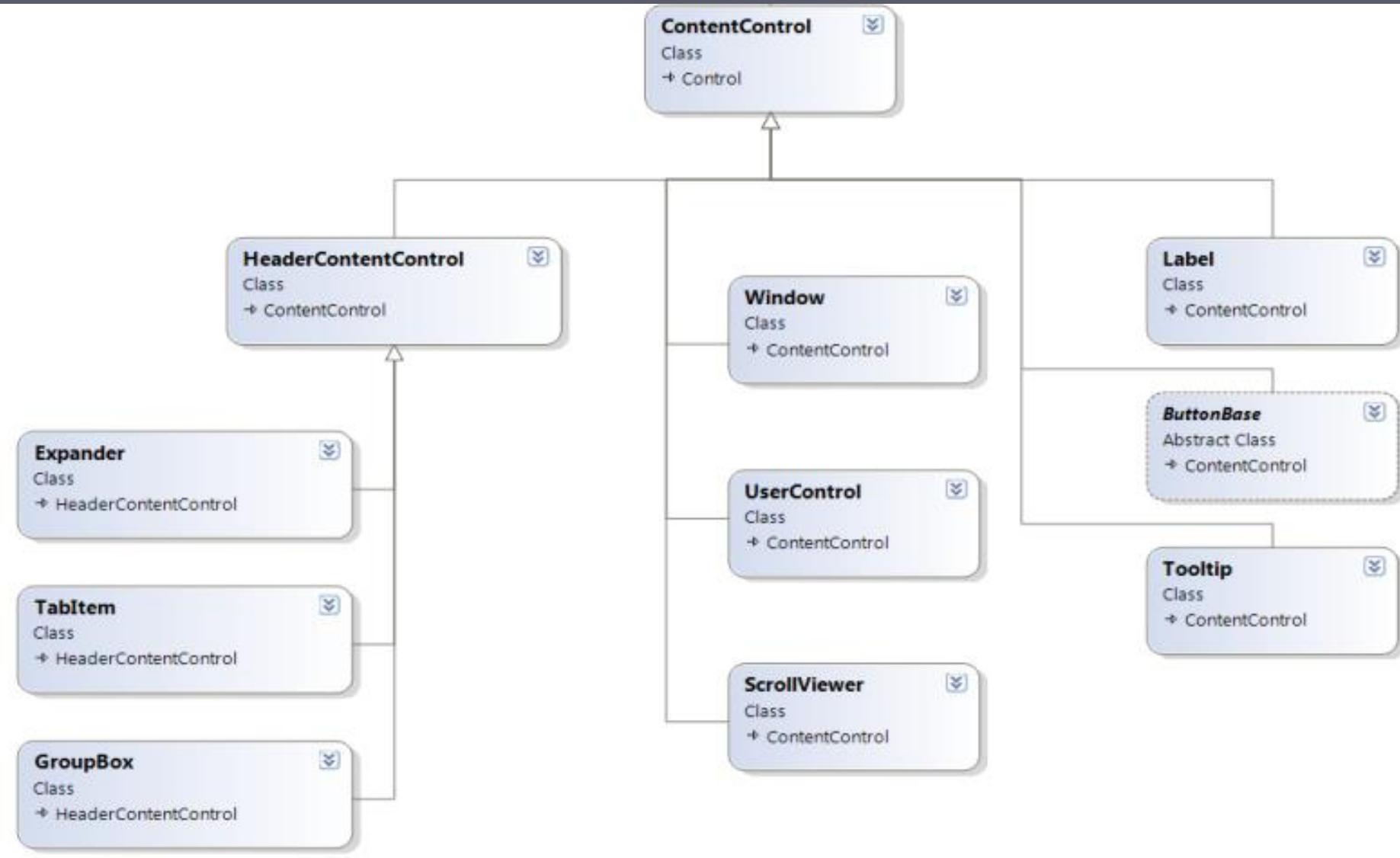


Content - одна порция
данных

коллекция каких-то
единиц информации

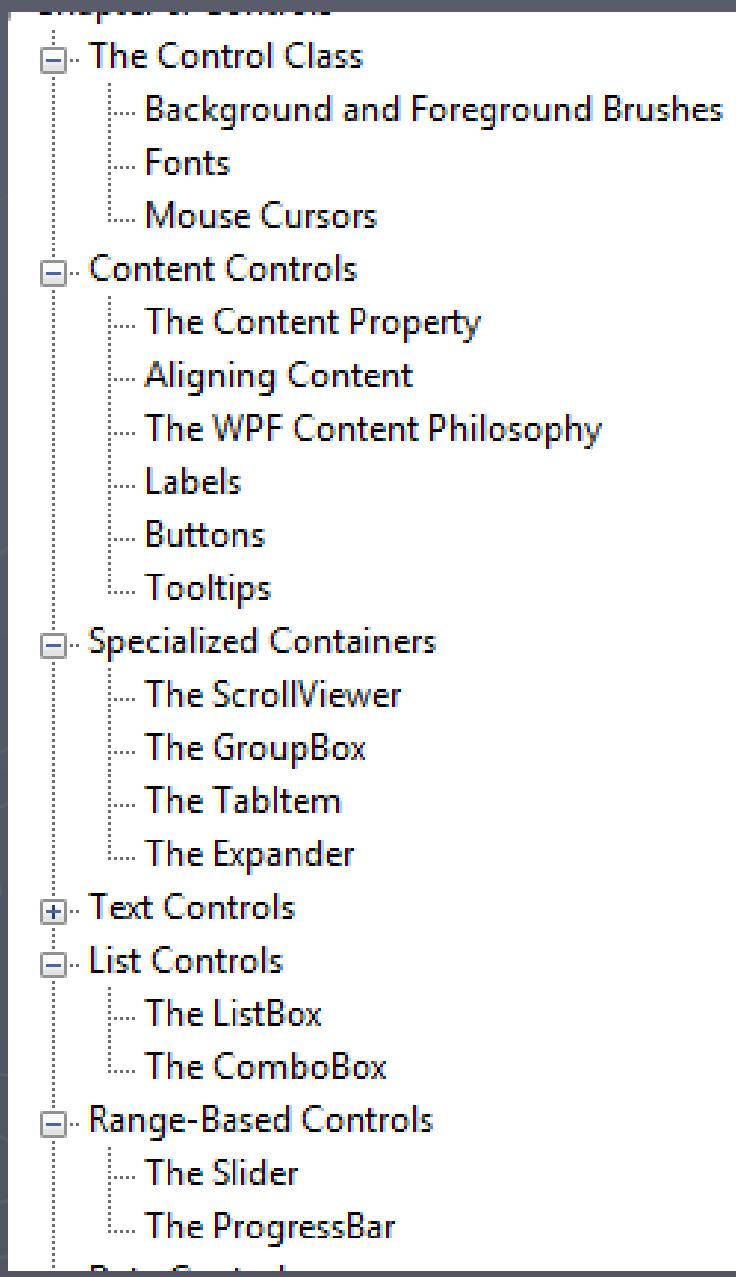
Объект класса, не
наследующего от
UIElement

Объект класса,
наследующего от
UIElement



ЭУ

- ▶ **Элементы управления содержимым** (Button Label)
- ▶ **Специальные контейнеры,** (ScrollViewer,GroupBox)
- ▶ **Декораторы** (Border или Viewbox)
- ▶ **Элементы управления списками** (ListBox, ComboBox)
- ▶ **Текстовые элементы управления,** (TextBox, RichTextBox)
- ▶ **Элементы, основанные на диапазонах значений**
(ProgressBar, Slider)
- ▶ **Элементы для работ с датами,** (DatePicker и Calendar)
- ▶ **Другие**



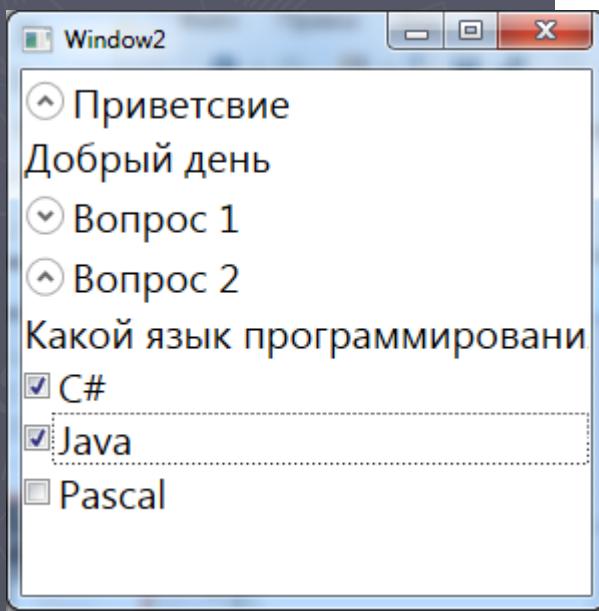
Новые ЭУ

RepeatButton

повторяется событие Click (свойства Delay и Interval)

Expander

Скрытое содержимое, раскрывающееся по нажатию на указатель (содержимое может быть разным)



```
<Expander Header="Вопрос 1"  
IsEnabled="True">  
  
<StackPanel>  
    <Expander Header="Приветствие">  
        <TextBlock>Добрый день</TextBlock>  
    </Expander>  
    <Expander Header="Вопрос 1">  
        <TextBlock>На каком вы курсе</TextBlock>  
    </Expander>  
    <Expander Header="Вопрос 2">  
        <StackPanel>  
            <TextBlock>Какой язык программирования вы изучаете</TextBlock>  
            <CheckBox>C#</CheckBox>  
            <CheckBox>Java</CheckBox>  
            <CheckBox>Pascal</CheckBox>  
        </StackPanel>  
    </Expander>
```

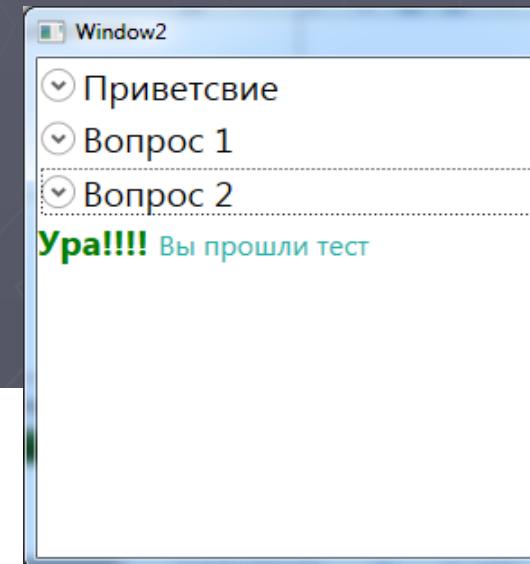
```
<Expander Header="Вопрос 2" Expanded="Expander_Expanded"  
Collapsed="Expander_Collapsed">
```

Открытие и закрытие

TextBlock

Блок текста с разным
форматированием

```
<TextBlock TextWrapping="Wrap">  
    <Run FontSize="20" Foreground="Green"  
          FontWeight="Bold">Ура!!!!</Run>  
    <Run FontSize="16" Foreground="LightSeaGreen">  
        Вы прошли тест</Run>  
</TextBlock>
```



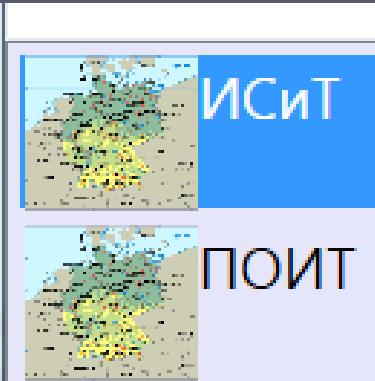
PasswordBox

не поддерживает работу с буфером обмена

```
<PasswordBox PasswordChar="*" MinHeight="30" />  
<PasswordBox MinHeight="30" />
```

ListBox

```
<StackPanel>  
    <ListBox Name="Photos" Background="Lavender">  
        <ListBoxItem Margin="3">  
            <StackPanel Orientation="Horizontal">  
                <Image Source="/files/map.gif" Width="60" />  
                <TextBlock>ИСиТ</TextBlock>  
            </StackPanel>  
        </ListBoxItem>  
        <ListBoxItem Margin="3">  
            <StackPanel Orientation="Horizontal">  
                <Image Source="/files/map.gif" Width="60" />  
                <TextBlock>ПОИТ</TextBlock></StackPanel>  
        </ListBoxItem>  
    </ListBox>  
</StackPanel>
```



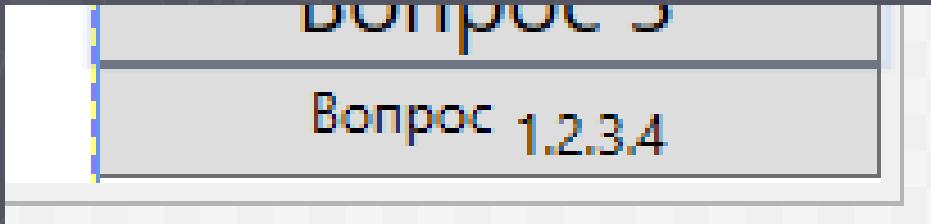
PopUp – сплывающее окно с содержимым

- 1) Имеет свойство PopupAnimation
- 2) Может содержать другие элементы управления
- 3) Может иметь прозрачность

```
<Popup Name="popLink"
       StaysOpen="False"
       Placement="Mouse"
       MaxWidth="200"
       PopupAnimation="Slide"
       AllowsTransparency = "True">
    <Border BorderBrush="Beige"
            BorderThickness="2"
            Background="White">
        <TextBlock Margin="10" TextWrapping="Wrap" >
            Нажмите кнопку
            <Button Content="OK"
                   Click="oK_Click"/>
            </TextBlock>
        </Border>
    </Popup>
```

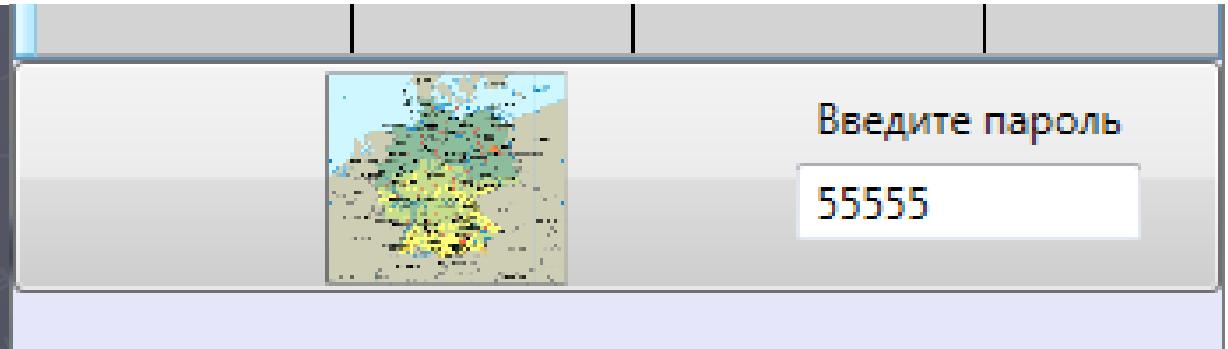
Вложенность компонентов

```
<Button>
    <Button.Content>
        <StackPanel Orientation="Horizontal">
            <TextBlock Text="Вопрос"/>
            <TextBlock Text="1.2.3.4" Margin="5,5,0,0"/>
        </StackPanel>
    </Button.Content>
</Button>
```



Вложение элементов

```
<StackPanel Orientation="Horizontal">
    <Image Source="files/map.gif"
        Height="60" Width="200" />
    <StackPanel Orientation="Vertical">
        <Label Content="Введите пароль"></Label>
        <TextBox> </TextBox>
    </StackPanel>
</StackPanel>
```



Tab

С какой стороны размещаются вкладки

```
<TabControl TabStripPlacement="Right">
    <TabItem>
        <TabItem.Header>
            <StackPanel>
                <Image Source="files/1.jpg" Height="70"
Width="70"/></Image>
                <TextBlock Text="Просмотр"/>
            </StackPanel>
        </TabItem.Header>
    </TabItem>
    <TabItem>
        <TabItem.Header>
            <StackPanel>
                <Image Source="files/2.jpg" Height="70"
Width="70"/></Image>
                <TextBlock Text="Изменить "/>
            </StackPanel>
        </TabItem.Header>
    </TabItem>

    <TabItem Header="ID">
    </TabItem>
</TabControl>
```

Вложенный
дескриптор



Просмотр

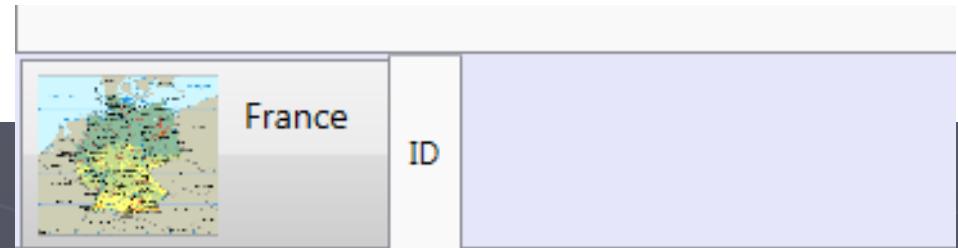


Изменить

ID

```
<TabControl >
  <TabItem>
    <TabItem.Header>
      <StackPanel Orientation="Horizontal">
        <Image Source="files/map.gif" Height="70<<
          Width="70"></Image>
        <TextBlock Margin="10" FontSize="14"
          HorizontalAlignment ="Center">France</TextBlock>
      </StackPanel>
    </TabItem.Header>
  </TabItem>

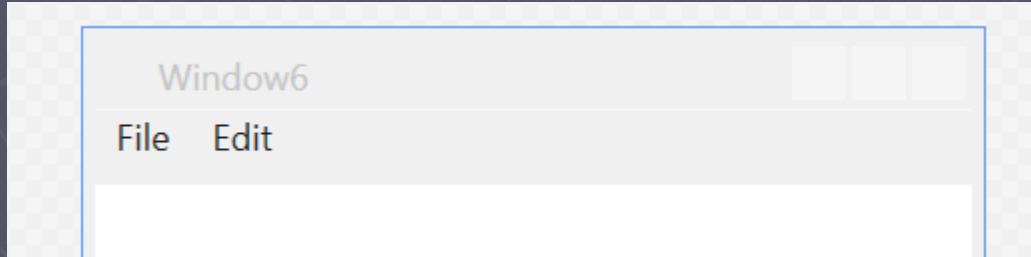
  <TabItem Header="ID">
    </TabItem>
  </TabControl>
```



Menu

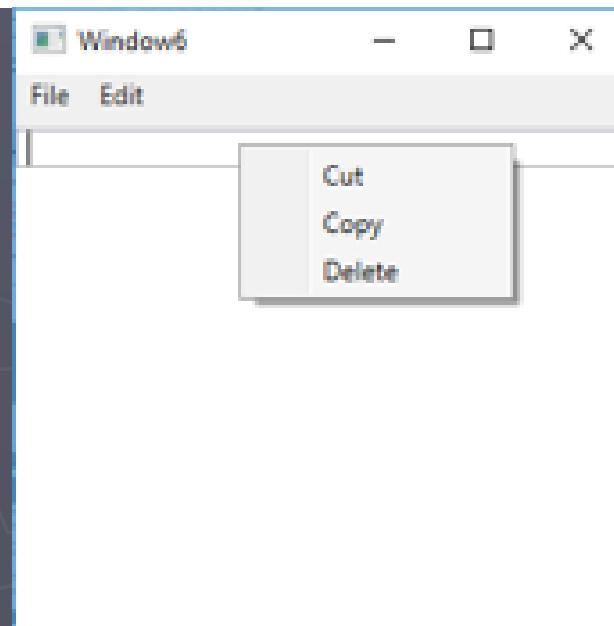
```
<Menu Height="25" VerticalAlignment="Top">
    <MenuItem Header="File">
        <MenuItem Header="New " ></MenuItem>
        <MenuItem Header="Open " >
            <MenuItem Header="Last"></MenuItem>
            <MenuItem Header="Prev" ></MenuItem>
        </MenuItem>
        <Separator />
        <MenuItem Header="Exit" ></MenuItem>
    </MenuItem>
    <MenuItem Header="Edit" ></MenuItem>

</Menu>
```



ContextMenu

```
<RichTextBox Name="Editor">
    <RichTextBox.ContextMenu>
        <ContextMenu>
            <MenuItem Header="Cut"></MenuItem>
            <MenuItem Header="Copy"></MenuItem>
            <MenuItem Header="Delete"></MenuItem>
        </ContextMenu>
    </RichTextBox.ContextMenu>
</RichTextBox>
```



- ▶ Элементы для работы с датами :
Calendar и **DatePicker**

- ▶ Панель инструментов **ToolBar**
- ▶ Элемент для работы с изображениями **Image**

Окна

- 1) События окна
- 2) Модальные (доступ к родителю запрещен ShowDialog()), не модельные (Show())
- 3) Управление и присоединение (owner)
- 4) Формы окон
 - 1) фоновая графика
 - 2) создать фон с векторным содержимым → среда Expression Blend
 - 3) элемент, имеющий необходимую форму → Border

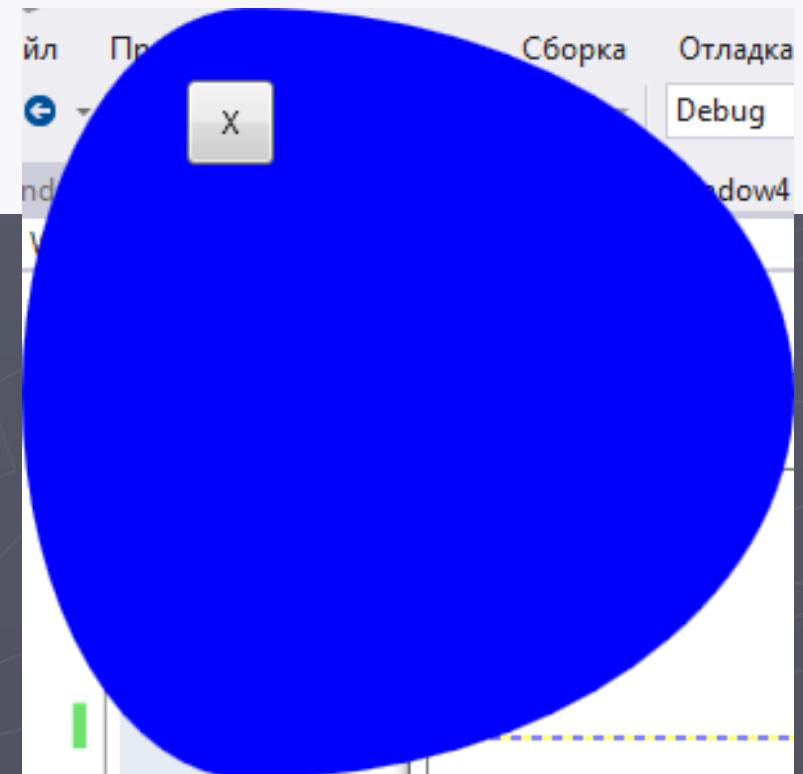
```
<Window x:Class="WpfAppDemo.Window5"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility"
    xmlns:local="clr-namespace:WpfAppDemo"
    mc:Ignorable="d"
    Title="Window5" Height="300" Width="300"
    AllowsTransparency="True"
    Background="Transparent"
    WindowStyle="None"
    MouseLeftButtonDown="Window5_OnMouseLeftButtonDown">
    <Border Background="Blue"
        CornerRadius="250, 600, 600, 250"
        >
        <Grid>
            <Button Content="X"
                Margin="64, 28, 202, 239"
                Name="ButtonX"
                Click="ButtonX_OnClick"/>
        </Grid>
    </Border>
</Window>
```

DecoratorBorder

- 1) свойства Window.AllowTransparency значение true.
- 2) WindowStyle → None, скрыть неклиентскую область окна
- 3) Установить для фона окна прозрачный цвет или задать изображение

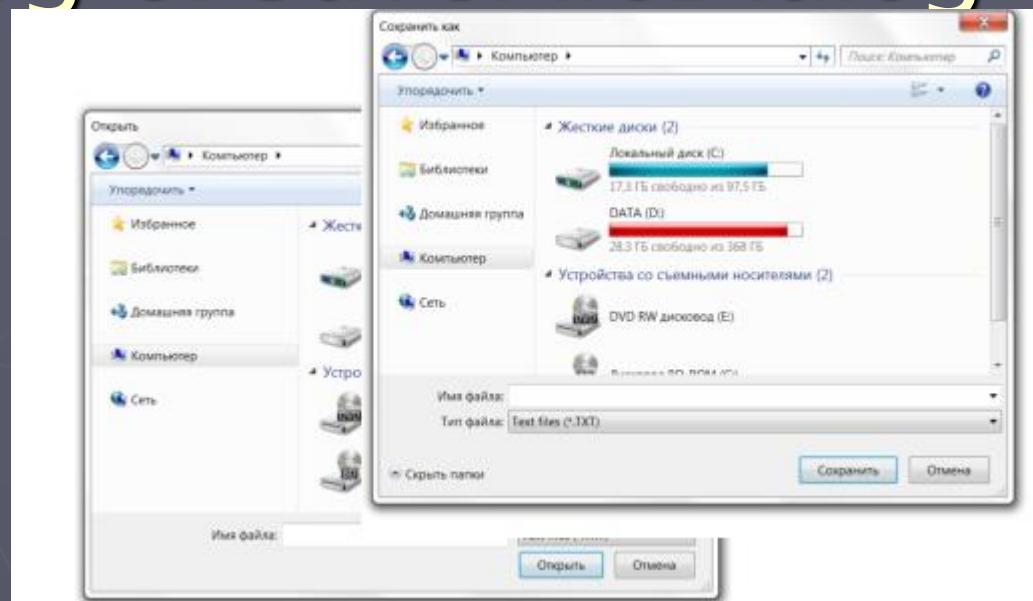
Окна не прямоугольной формы

```
private void Window5_MouseLeftButtonDown(object sender,  
MouseButtonEventArgs e)  
{  
    DragMove();  
}  
  
private void ButtonX_OnClick(object sender,  
RoutedEventArgs e)  
{  
    App.Current.Shutdown();  
}
```



OpenFileDialog & SaveFileDialog

► Microsoft.Win32

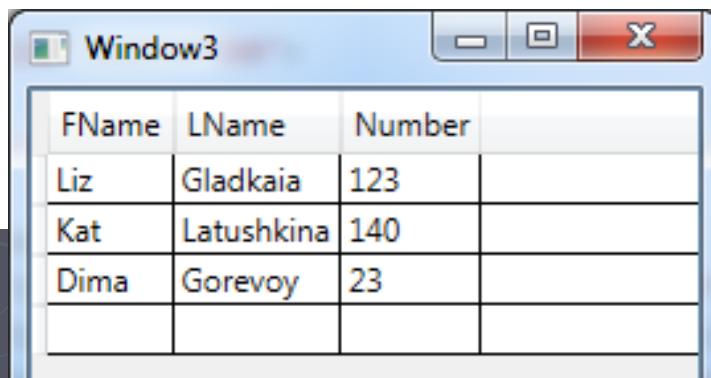


► DataGrid

```
<Window x:Class="WpfAppDemo.Window3"
    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/2009/design"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:WpfAppDemo"
    mc:Ignorable="d"
    Title="Window3" Height="500" Width="500"
    xmlns:col="clr-namespace:System.Collections;assembly=mscorlib">
<
<Grid Background="Lavender">
    <DataGrid x:Name="SrudGrid" AutoGenerateColumns="True"
        ItemsSource="{DynamicResource ResourceKey=fit}">
        <DataGrid.Resources>
            <col:ArrayList x:Key="fit">
                <local:Student FName="Liz" LName="Gladkaia" Number="123" />
                <local:Student FName="Kat" LName="Latushkina" Number="140" />
                <local:Student FName="Dima" LName="Gorevoy" Number="23" />
            </col:ArrayList>
        </DataGrid.Resources>
    </DataGrid>
</Grid>
</Window>
```

```
namespace WpfAppDemo
{
    public class Student
    {
        public String FName { get; set; }
        public String LName { get; set; }
        public String Number { get; set; }
        public override string ToString()
        {
            return FName + LName+
Number.ToString();
        }
    }
}
```

автоматически
разбивать на
столбцы



| FName | LName | Number | |
|-------|------------|--------|--|
| Liz | Gladkaia | 123 | |
| Kat | Latushkina | 140 | |
| Dima | Gorevoy | 23 | |

```

public partial class Window3 : Window
{
    public Window3()
    {
        InitializeComponent();
        List<Student> studList = new List<Student>
        {
            new Student { FName= "Eugen", LName= "Riabchenko", Number= 89 },
            new Student { FName= "Fedor", LName = "Plehanov", Number= 367 },
        };
        SecondStudGrid.ItemsSource = studList;
    }
}

```

```

<Grid Background="Lavender">
    <StackPanel Orientation="Vertical">

        <DataGrid x:Name="SrudGrid" AutoGenerateColumns="True" ItemsSource="{DynamicResource ResourceKey=fit}">
            ...
        </DataGrid>

        <DataGrid Name="SecondStudGrid" AutoGenerateColumns="True">
        </DataGrid>

    </StackPanel>
</Grid>

```

| FName | LName | Number |
|-------|------------|--------|
| Liz | Gladkaia | 123 |
| Kat | Latushkina | 140 |
| Dima | Gorevoy | 23 |
| | | |

| FName | LName | Number |
|-------|------------|--------|
| Eugen | Riabchenko | 89 |
| Fedor | Plehanov | 367 |
| | | |

Дополнительные свойства

```
<DataGrid Name="TherdStudGrid" AutoGenerateColumns="False"
          HorizontalGridLinesBrush="DarkGray"
          RowBackground="LightGray"
          AlternatingRowBackground="White">
    <DataGrid.Columns>
        <DataGridTextColumn Header="ID"
            Binding="{Binding Path=Number}" Width="90" />
        <DataGridHyperlinkColumn Header="Имя"
            Binding="{Binding Path=FName}" Width="80" />
        <DataGridTextColumn Header="Фамилия"
            Binding="{Binding Path=LName}" Width="100" />
    </DataGrid.Columns>
</DataGrid>
```

Изменение цвета фона

Определение столбцов и заголовков

Возможность ввода текста

Гиперссылка

| ID | Имя | Фамилия |
|------|-------|------------|
| 8923 | Eugen | Riabchenko |
| 367 | Fedor | Plehanov |
| | | |

- `DataGridViewColumn` - позволяет задать специфичный шаблон для отображения столбца

```
</DataGrid.Columns>
```

детали строк (row details)

```
    <DataGrid.RowDetailsTemplate>
```

```
        <DataTemplate>
```

```
            <StackPanel Orientation="Horizontal">
```

```
                <Image Source="files/map.gif" Height="30"  
                      Width="40"> </Image>
```

```
                <TextBlock Text="{Binding Path=LName}" />
```

```
                <TextBlock Text="{Binding Path=FName}" />
```

```
                <TextBlock Text=" ID = " />
```

```
                <TextBlock Text="{Binding Path=Number}" />
```

```
            </StackPanel>
```

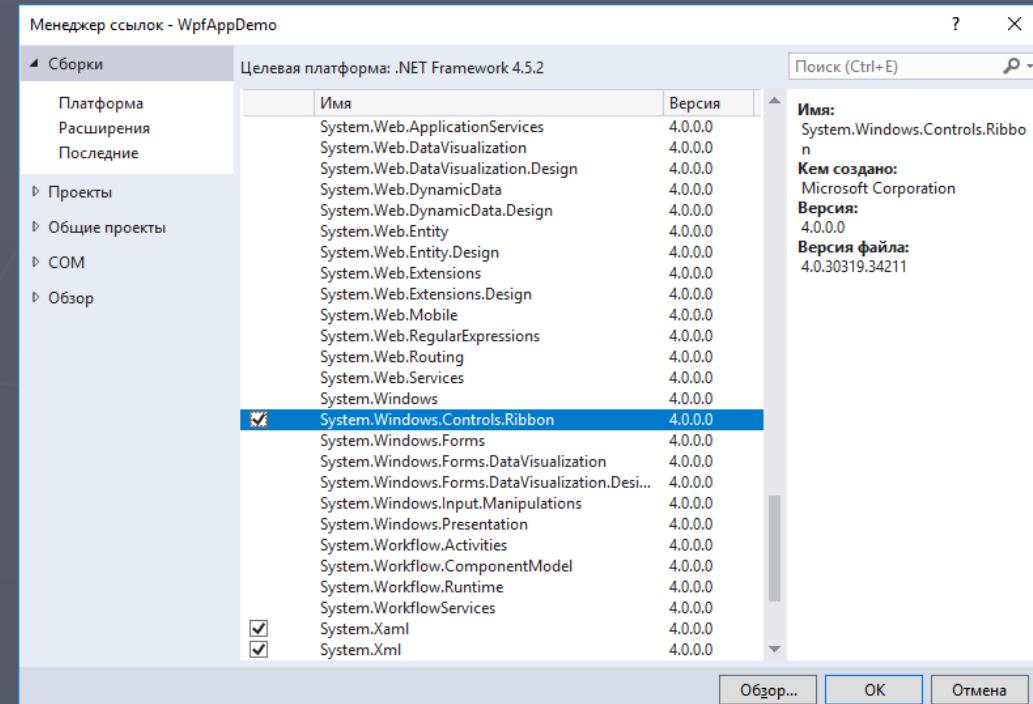
```
        </DataTemplate>
```

```
    </DataGrid.RowDetailsTemplate>
```

```
</DataGrid>
```

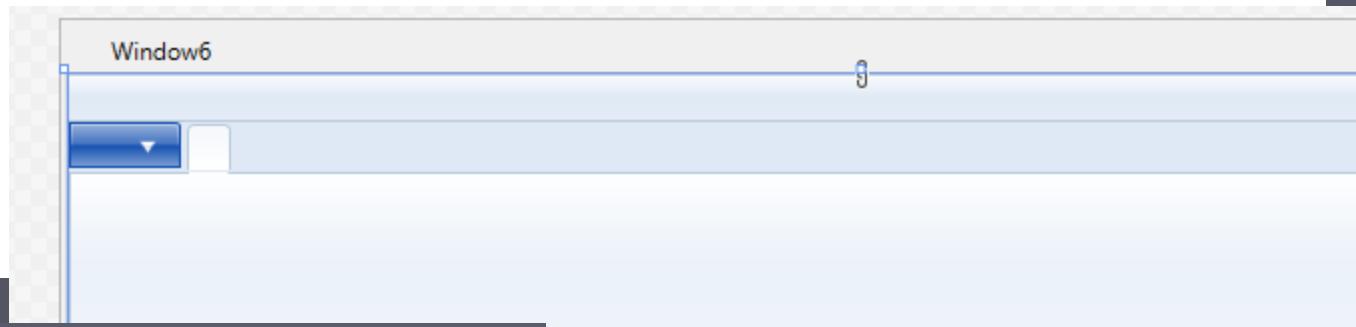
| ID | Имя | Фамилия |
|-----|------------------------|------------|
| 39 | Eugen | Riabchenko |
| 367 | Fedor | Plehanov |
| | PlehanovFedor ID = 367 | |

Лента (Ribbon)



```
<Window x:Class="WpfAppDemo.Window6"
    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:WpfAppDemo"
    xmlns:ribbon="clr-
namespace:System.Windows.Controls.Ribbon.Primitives;assembly=System.Windows.Con
trols.Ribbon"
    mc:Ignorable="d"
    Title="Window6" Height="450" Width="800">
<Grid>
    <DockPanel>
        <Ribbon>
            <ribbon:RibbonTabsPanel></ribbon:RibbonTabsPanel>

        </Ribbon>
    </DockPanel>
</Grid>
</Window>
```



- панели для быстрого запуска
- меню приложения
- лента с вкладками



.NET

APIs

.NET Core

.NET Framework

ASP.NET

Xamarin

Azure

Фильтровать по названию

Элементы управления

Категории элементов управления

Модель содержимого WPF

› Библиотека элементов управления

› Стили и шаблоны

› Настройка элементов управления

Элементы управления

30.03.2017 • Время чтения: 10 мин • Соавторы

Windows Presentation Foundation (WPF) содержит большинство распространенных компонентов пользовательского интерфейса, которые используются практически во всех приложениях Windows, таких как [Button](#), и [ListBox](#). Исторически эти объекты называются элементами управления. Хотя WPF SDK продолжает использовать термин «элемент управления» для общего обозначения любого класса, который представляет видимый элемент в приложении, важно отметить, что класс не обязательно должен наследовать от [Control](#) класса, чтобы представлять элемент управления. Классы, наследующие от [Control](#) класса, должны содержать [ControlTemplate](#), позволяющий гибко изменять внешний вид элемента управления, не создавая новую модель. В этом разделе обсуждаются как элементы управления (наследующих или наследующих от [Control](#) класса), так и возвращающие значение элементы, обычно используемые в WPF.

Создание экземпляра элемента управления

Элемент управления можно добавить в приложение с помощью Языка XAML или кода. В следующем примере показано, как создать простое приложение, которое запрашивает у пользователя имя и фамилию. В этом примере используется один из самых распространенных способов создания элементов управления: два метки, два текстовых поля и две кнопки в XAML. Все элементы управления могут быть созданы аналогичным образом.

XAML

```
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition Height="30"/>
        <RowDefinition Height="30"/>
```