WPF (Windows Presentation Foundation)

Last Update - 2023.03.25

www.albdarb.com
www.highcode.am
Albert S. Darbinyan
"HighCode" LLC
French University in Armenia (UFAR)
Gavar State University (GSU)

……...…….

Lab1
- **WPF Introduction** (Համառոտ ներածություն) [Мак-Дональд -20]
- **XAML** [Мак-Дональд -48; Натан- 43]
- **Layout** (Grid, StackPanel, WrapPanel, DockPanel, Canvas) [Мак-Дональд -81; Натан -146]

Lab2
- **Content controls**
  Button [Мак-Дональд -183; Натан –305]
  TextBox [Мак-Дональд -202; Натан -362]
  ListBox [Мак-Дональд -207; Натан – 332]
  RadioButton [Мак-Дональд -186; Натан -309]
  Slider [Мак-Дональд -211; Натан - 385]
  ChekBox [Мак-Дональд -186; Натан –308]

Lab3
- **Graphics 2D** [Мак-Дональд – 324; Натан – 533]
- **Path, Geometry** [Мак-Дональд -361]

Lab4
- **Bezier** [Мак-Дональд -370; Петцольд -817]

Lab5
- **Images** [Мак-Дональд –376 ; Натан – 356]
- **ZAR**
- **Play Chess**

Lab6
- **Menu** [Мак-Дональд 780; Натан – 298eng]
- **Dialog** [Мак-Дональд -696]

Lab7
- **File Read/Write** [Нейгел2010 -893rus; Nagel2012-673]
- **ToolBar** [Мак-Дональд -786; Натан – 351]

Lab8
- **TreeView** [Мак-Дональд -672; Натан – 302]

Lab9
- **Sound** [Мак-Дональд -805 ; Натан – 722]
- **Video** [Мак-Дональд -805 ; Натан – 722]

Lab10
- **Drawing 3D** [Мак-Дональд-827; Натан -602]

Lab11
- **Animation** [Мак-Дональд -402; Натан – 675]

Lab12
- **3D Animation** [Мак-Дональд -852]

Պատկերների համար ծածկագիրը user: **file** password: **file**

ԳՐԱԿԱՆՈՒԹՅՈՒՆ

www.albdarb.com User: books Psw: "wpf" Psw: "wpfeng"

1. Мак-Дональд М. –" Pro WPF in Csharp 2010", 2011.
2. Натан Адам – "WPF4, Подробное руководство", 2011.
3. Троелсен Эндрю … –"Язык программирования C# 7 и платформы .NET и .NET Core",2018.
4. Rod Stephens – "WPF Programmer's Reference: C# 2010.NET4", 2010.
5. https://metanit.com/sharp/

- **WPF** (Windows Presentation Foundation)

Lab1 :

Համառոտ ներածություն [Мак-Дональд -20]

- WPF – ը իրենից ներկայացնում է `visual` ծրագրավորման գրաֆիկական միջոց, որի հիմքում ընկած է DirectX գրաֆիկական արագացուցիչի օգտագործումը: Այն հնարավորություն է տալիս օգտվել համակարգչի գրաֆիկական պրոցեսորների միջոցներից, ապահովելով բարձր արտադրողականություն: Գրաֆիկական այդպիսի հնարավորությունը բացակայում էր Windows.Form -ում:
- WPF – ում կիրառվում է Web ծրագրավորման դեկլարատիվ տեգային ներկայացումը XAML համակարգով: Այն չի թարգմանվում տրամաբանական կոդի հետ, այլ սկզբից սերիալիզացիա է արվում XAML ի դիսկրիպտորում, կատարվում է գրաֆիկական ինտերֆեյսի օբյեկտների գեներացիա եւ վերջին հաշվով վերածվում է `C#` տրամաբանական կոդի: [Мак-Дональд -47]
- WPF – ում անիմացիա, աուդիո, վիդեո համակարգերը իրականացված են նոր մակարդակով:
- WPF – ում ներդրված է 3D գրաֆիկական համակարգ:
- WPF –ից առաջ C#.NET ծրագրային համակագը օգտվում էր Win32 API – ից, որը ներառում է User32 էլեմենտներ և GDI գրաֆիկական ներկայացում: WPF –ը ներկայացնում է գրաֆիկական պլատֆորմ իր նոր գրադարաններով: WPF -ում գրաֆիկական մասը առանձնացվում է տրամաբանական կոդից, այս ձևով ապահովելով ցուցադրական եւ տրամաբանական մասերի անկախությունը:

// ……

XAML [Мак-Дональд -48; Натан- 43]

- WPF ում ծրագրի և դիզայներական մասը և տրամաբանությունը կարելի է գրել C# ի կոդերով: Սակայն քայլելով ժամանակի հետ, թույլատրվում է նաև դիզայներական մասը գրել տեգերի միջոցով, որը կոչվում է XML մաս (հնչում է “զզամլ”): Տեգերը ունեն ատրիբուտներ, որոնք թարգմանությունից հետո դառնում են C# ի հատկանիշներ: Տեգերում ցանկալի է նաև իրականացնել ծրագրի ղեկավարման էլեմենտների իրադարձությունները:

```
//xaml
<Window x:Class="WpfApplication12.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow albert" Height="350" Width="525">
    <Grid Background="green" MouseDown="Grid_MouseDown_1" Margin="111,10,22,22">
    </Grid>
</Window>
//xaml.cs
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }
        void Grid_MouseDown_1(object sender, MouseButtonEventArgs e)
        {
            MessageBox.Show("barev");
        }
    }
```

- `x:Class` -ով հայտարարվում է կլաս, որտեղից սկսվում է աշխատանքը:
- `xmlns = ""`-ով նշվում է անունի տարածք, որից ընդհանուր ձևով օգտվում են տեգերը:
- `xmlns:x= ""`-ով նշվում է անունի տարածք, որին կարելի է դիմել x= անունով նշելով միայն տվյալ անունի տարածքը:
- Ամբողջ ցուցադրման տարածքը ղեկավարվում է `<Window> </Window>` տեգով, որտեղ կարելի է տեղակայել էլեմենտներ, կոնտեյներներ և այլ միջոցներ:

Layout – (Grid, StackPanel, WrapPanel, DockPanel, Canvas)
[Мак-Дональд -81; Натан -146]

- WPF ում որպես պատուհանի ներկայացման պատասխանատու հանդես է գալիս Windows կլասը(Windows Form ում այդ դերը կատարում է Form կլասը), որից օգտվում են ժառանգականությամբ: Ցանկալի է Widows պատուհանում տեղադրել կոնտեյներ (Grid, StackPanel, WrapPanel, DockPanel, Canvas) և աշխատել նրանց միջոցով:
- Ցանկալի է էլեմենտների չափեր եւ տեսքը ղեկավարել Layout –ներով:
- Գործում են կոնտեյներում էլեմենտների տարբեր տեղակայման հատկանիշներ, որոնք օգնում են ցուցադրմանը:

✓ `Height="" Width="" HorizontalAlignment="" VerticalAlignment="" Margin=""`

```
//-------------------------
```

❖ **Grid**

Grid –ով հնարավորություն է տրվում, որպեսզի պատուհանի ցուցադրող մասը ստանա աղյուսակի տեսք: Եթե աղյուսակի տողերը եւ սյուները չեն նշված, ապա ցուցադրող մասը ունի մեկ վանդակ ընդգրկելով ամբողջ պատուհանը:

```
//xaml
<Window x:Class="WpfApplication16.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525">
    <Grid ShowGridLines="True">
        <Grid.RowDefinitions>
            <RowDefinition></RowDefinition>
            <RowDefinition></RowDefinition>
            <RowDefinition></RowDefinition>
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Button Grid.Row="2" Grid.Column="1" Margin="10,20,100,10">1</Button>
        <Button Click="Button_Click_1">2</Button>
        <Button Grid.Row="1" Grid.Column="0" Grid.ColumnSpan="2">Span Button</Button>
    </Grid>
</Window>
//xaml.cs
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }
        void Button_Click_1(object sender, RoutedEventArgs e)
        {
            MessageBox.Show("barev");
        }
    }
//--------------------------------
```

❖ **StackPanel**

- StackPanel -ով հնարավորություն է տրվում պատուհանի ցուցադրող մասը ընդգրկի էլեմենտներ ստեկի ձևով, կամ վրևից ներքև կամ ձախից աջ:

```
//xaml
<Window x:Class="WpfApplication13.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http: //schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525">
    <StackPanel Orientation="Vertical">
       <Button Background="Bisque" Width="100" Height="100">1</Button>
       <Button>2</Button>
       <Label>barev</Label>
       <Button MaxWidth="600">3</Button>
    </StackPanel>
</Window>
//---------------------------------
```

❖ **WrapPanel**

- WrapPanel -ով հնարավորություն է տրվում պատուհանի ցուցադրող մասը դասավորվի ըստ հերթականության մինչև պատուհանը հետևելով ձախից աջ վերևից ներքև:

```
//xaml
    <WrapPanel>
        <Button Width="222" Height="111">1</Button>
        <Button Width="333" Height="50">2</Button>
    </WrapPanel>
//-------------------------------------------------------------
```

❖ **DockPanel**

- DockPanel -ով էլեմենտները ձգտում են հարմարվել իրենց հատկացված տեղին:

```
//xaml
    <DockPanel LastChildFill="True">
        <Button DockPanel.Dock="Top" Background="Red" Content="TOP" />
        <Button DockPanel.Dock="Bottom" Background="Blue" Content="BOTTOM" />
        <Button DockPanel.Dock="Left" Background="Aquamarine" Content="LEFT" />
        <Button DockPanel.Dock="Right" Background="Green" Content="RIGHT" />
        <Button Background="LightGreen" Content="CENTER" />
    </DockPanel>
```

`LastChildFill="True"` -ն հնարավորություն է տալիս զբաղեցնել մնացած տարածքը ամբողջությամբ:

```
//-------------------------------------------------------------
```

❖ **Canvas**

- Canvas -ով հնարավորություն է տրվում պատուհանի ցուցադրող մասում էլեմենտները տեղադրվեն **ցանկացած** կոորդինատով եւ `Canvas.Zindex` հրամանով որոշի ծածկելու առավելությունը:

```
//xaml
<Canvas>
        <Button Width="50" Height="50" Canvas.ZIndex="1">2</Button>
        <Button Canvas.Left="10" Width="111" Height="111" Canvas.ZIndex="0">1</Button>
</Canvas>
```

……

Lab2 :

Ղեկավարման էլեմենտներ

- Հաշվել ֆակտորիալ և ֆիբոնաչի

Button [Мак-Дональд -183; Натан –305]

TextBox [Мак-Дональд -202; Натан -362]

ListBox [Мак-Дональд -207; Натан – 332]

```
// Factorial
<Grid>
        <Button Width="100" Height="50" Content="Factorial" HorizontalAlignment="Left"
                VerticalAlignment="Bottom" Margin="10,0,0,10" Click="Button_Click">
        </Button>
        <TextBox Name="txtmutq"  Width="100" Height="25" HorizontalAlignment="Left"
                 VerticalAlignment="Top">15</TextBox>
        <TextBox Name="txtelq" Width="100" Height="200" HorizontalAlignment="Left"/>
        <ListBox Name="elqlist" Width="100" Height="200"  HorizontalAlignment="Right"/>
        <TextBlock Name="elqblock" Width="100" Height="200" Background="Aqua"></TextBlock>
        <Label Width="100" Height="50" Margin="0,-200,0,0">elqblock</Label>
</Grid>
public partial class MainWindow : Window
{
        public MainWindow()
        {
            InitializeComponent();
        }
        void Button_Click(object sender, RoutedEventArgs e)
        {
            // txtelq.Text = "";
            // elqblock.Text = "";
            // elqlist.Items.Clear();
            int a = Convert.ToInt32(txtmutq.Text);
            long fact = 1;
            for (int i = 1; i <= a; i++)
            {
                fact = fact * i;
                txtelq.Text += Convert.ToString(fact) + "\n";
                elqblock.Text += Convert.ToString(fact) + "\n";
                elqlist.Items.Add(fact);
            }
            //  MessageBox.Show(fact.ToString());
        }
}
//-----------------------------------------
❖ Տնային (ֆիբոնաչի շարք 1 1 2 3 5 8 13 21 34 55 ...)
//-----------------------------------------
```

- Ստեղծել Button C# կոդով

```
//xaml
<Grid x:Name="bt">
</Grid>
//cs
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
            Button myButton = new Button();
            myButton.Width = 100;
            myButton.Height = 30;
```

```
            myButton.Content = "Barev      bolorin";
            bt.Children.Add(myButton);
            myButton.Click += MyButton_Click;
        }
    }
//-----------------------------------------------
<Button Content="&lt;&quot; Hello&quot;&gt;"/>
<Button xml:space="preserve">
    Hello         World
</Button>
//-----------------------------------------------
```

RadioButton [Мак-Дональд -186; Натан -309]

- Ընտրել RadioButton –ով էլիպսի գույները

```
//xaml
    <Grid>
        <RadioButton Name="red" Margin="10,10,0,0" Click="red_Click" >  karmir
        </RadioButton>
        <RadioButton Name="green" Margin="10,50,0,0" Click="green_Click"> kanach
        </RadioButton>
        <RadioButton Name="blue" Margin="10,100,0,0" Click="blue_Click"> kapuyt
        </RadioButton>
        <Ellipse Name="el" Fill="Yellow" Width="200" Height="100" >
        </Ellipse>
    </Grid>
//cs
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }
        void red_Click(object sender, RoutedEventArgs e)
        {
            el.Fill = Brushes.Red;
        }
        void green_Click(object sender, RoutedEventArgs e)
        {
            el.Fill = Brushes.Green;
        }
        void blue_Click(object sender, RoutedEventArgs e)
        {
            el.Fill = Brushes.Blue;
            // el.Fill = new SolidColorBrush(Color.FromRgb(111,111,111));
        }
    }
//-----------------------------------------------
```

Slider [Мак-Дональд -211; Натан - 385]

- Կատարել Slider –ով ուղղանկյան չափերի փոփոխություն

```
<Grid>
      <Rectangle Name="rec" Margin="200,20,0,0" Width="100" Height="200" Fill="Bisque"
      VerticalAlignment="Top"></Rectangle>
      <Slider Name="sl" VerticalAlignment="Bottom" Width="300" Height="30" Margin="0,0,0,10"
      Background="AliceBlue"  Value="100" Maximum="200" ValueChanged="sl_ValueChanged_1">
      </Slider>
</Grid>
```

```
public partial class MainWindow : Window
{
    public MainWindow()
    {
        InitializeComponent();
    }
    void sl_ValueChanged_1(object sender, RoutedPropertyChangedEventArgs<double> e)
    {
        rec.Width = sl.Value;
        rec.Height = sl.Value;
        //rec.Margin =new Thickness(10,10,10,10);
    }
}
//-----------------------------------------------
```

ChekBox [Мак-Дональд -186; Натан –308]

- Կատարել CheckBox -ով տեքստի **Bold**, *Italic* և Underline տարբեր կոմբինացիաներ:

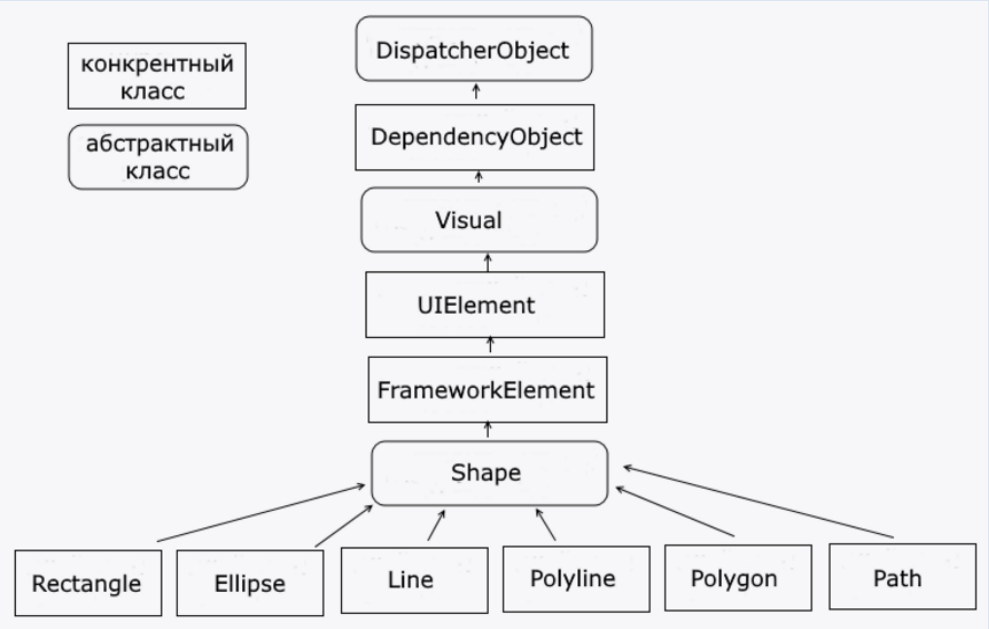
Օրինակում **bold** և **italic** կազմակերպվում է Checked և Unchecked իրադարձություններով, իսկ **uderline** իրականացվում է Click իրադարձությունով:

```
    <Grid>
        <CheckBox Name="bold" Width="100" Margin="10,0,400,300" Checked="bold_Checked"
         Unchecked="bold_Unchecked">Bold</CheckBox>
        <CheckBox Name="italic" Width="100" Margin="10,50,400,250"
         Checked="italic_Checked" Unchecked="italic_Unchecked">Italic</CheckBox>
        <CheckBox Name="underline" Width="100" Margin="10,100,400,200"
         Click="CheckBox_Click">Uderline</CheckBox>
        <TextBlock Name="tb" Width="200" Height="400" FontSize="60">barev</TextBlock>
    </Grid>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }
        void bold_Checked(object sender, RoutedEventArgs e)
        {
            tb.FontWeight = FontWeights.Bold;
        }
        void bold_Unchecked(object sender, RoutedEventArgs e)
        {
            tb.FontWeight = FontWeights.Normal;
        }
        void italic_Checked(object sender, RoutedEventArgs e)
        {
            tb.FontStyle = FontStyles.Italic;
        }
        void italic_Unchecked(object sender, RoutedEventArgs e)
        {
            tb.FontStyle = FontStyles.Normal;
        }
        void CheckBox_Click(object sender, RoutedEventArgs e)
        {
            if ((bool)underline.IsChecked)
                tb.TextDecorations = TextDecorations.Underline;
            else
                tb.TextDecorations = null;
        }
    }
```

Lab3:

Graphics 2D

[Мак-Дональд – 324; Натан – 533]



- Գրել մատիտի ծրագիր, որը փոխարինի <InkCanvas> </InkCanvas> տեգին:

```
//xaml
<Grid x:Name="gg">
    <!--<InkCanvas></InkCanvas>-->
</Grid>
//cs
public partial class MainWindow : Window
{
    bool b;
    double x;
    double y;
    public MainWindow()
    {
        InitializeComponent();
        MouseDown += MainWindow_MouseDown;
        MouseMove += MainWindow_MouseMove;
        MouseUp += MainWindow_MouseUp;
    }
    void MainWindow_MouseUp(object sender, MouseButtonEventArgs e)
    {
        b = false;
        Mouse.Capture(null);
        //ReleaseMouseCapture();
    }
    void MainWindow_MouseMove(object sender, MouseEventArgs e)
    {
        Line a = new Line();
        //gg.Children.Remove(a);
        //gg.Children.Clear();
        if (b)
        {
            gg.Children.Add(a);
            a.Stroke = Brushes.Black;
            a.StrokeThickness = 6;
            a.X1 = x;
            a.Y1 = y;
```

```
            a.X2 = e.GetPosition(this).X;
            a.Y2 = e.GetPosition(this).Y;
        }
        x = a.X2;
        y = a.Y2;
    }
    void MainWindow_MouseDown(object sender, MouseButtonEventArgs e)
    {
        b = true;
        x = e.GetPosition(this).X;
        y = e.GetPosition(this).Y;
        Mouse.Capture(this);
        //CaptureMouse();
    }
}
//.....................................................
```

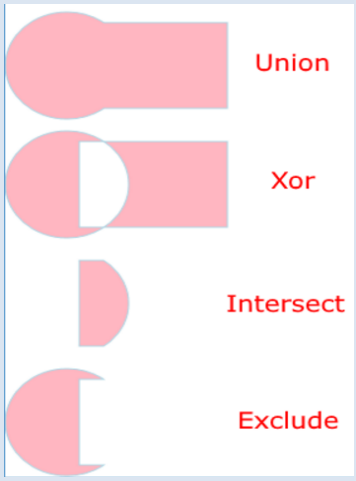
Path, Geometry

[Мак-Дональд -361]

- Բարդ ֆիգուրաներ եւ նրանց կոմբինացիաներ կարելի է ստանալ `Path` տեգի միջոցով: Այն ունի `Data` հատկանիշ, որը որոշում է `Geometry` օբյեկտներ: `Geometry` կլասը աբստրակտ է: `Path` կլաը ունի հետևյալ ժառանգ կլասները.

✓ **LineGeometry** - գիծ: **RectangleGeometry** - ուղղանկյուն: **EllipseGeometry** – էլիպս: **PathGeometry** - պարզ ֆիգուրաներից ստանում է բարդ կոմբինացիա: **GeometryGroup** - `Geometry` օբյեկտների խմբավորում:

- **CombinedGeometry** -ը կատարում է երկու `Geometry` օբյեկտների խմբավորում: Այն տարբերվում է `GeometryGroup` -ից նրանով, որ ունի `GeometryCombinneMode` հատկանիշ, որով խմբավորվում են երկու ֆիգուրաներ տարբեր հատումներով: `Union; Intersect; Xor; Exclude`



- Կատարել էլիպսի եւ ուղղանկյան միավորում եւ մկնիկի միջոցով տեղաշարժ:

```
//xaml
    <Grid>
        <Path HorizontalAlignment="Left" VerticalAlignment="Top" Name="MyPath" Fill="Yellow"
          Stroke="Black" MouseDown="MyPath_MouseDown" MouseMove="MyPath_MouseMove"
          MouseUp="MyPath_MouseUp">
          <Path.Data>
                <CombinedGeometry GeometryCombineMode="Union">
                    <CombinedGeometry.Geometry1>
                        <RectangleGeometry Rect="0,0,150,150"></RectangleGeometry>
                    </CombinedGeometry.Geometry1>
                    <CombinedGeometry.Geometry2>
```

```
                        <EllipseGeometry Center="80,80" RadiusX="110" RadiusY="50">
                        </EllipseGeometry>
                    </CombinedGeometry.Geometry2>
                </CombinedGeometry>
            </Path.Data>
        </Path>
    </Grid>
//cs
public partial class MainWindow: Window
{
    bool t;
    double x, y;
    public MainWindow()
    {
        InitializeComponent();
    }
    void MyPath_MouseDown(object sender, MouseButtonEventArgs e)
    {
        t = true;
        x = e.GetPosition(this).X - MyPath.Margin.Left;
        y = e.GetPosition(this).Y - MyPath.Margin.Top;
    }
    void MyPath_MouseMove(object sender, MouseEventArgs e)
    {
        if (t)
        {
            MyPath.Margin = new Thickness(e.GetPosition(this).X - x,
                                e.GetPosition(this).Y - y, 0, 0);
        }
    }
    void MyPath_MouseUp(object sender, MouseButtonEventArgs e)
    {
        t = false;
    }
}
//-------------------------------------
```

- `<GeometryGroup>` Հնարավորություն է տալիս երկուսից ավել ֆիգուրաներ ընդգրկել

```
    <Path Stroke="Black">
      <Path.Data>
                <GeometryGroup>
                    <RectangleGeometry Rect="0,0,150,150"></RectangleGeometry>
                    <EllipseGeometry Center="80,80" RadiusX="110" RadiusY="50">
                    </EllipseGeometry>
                    <PathGeometry>
                        <PathFigure IsClosed="True" StartPoint="10,100">
                            <LineSegment Point="100,100"/>
                            <LineSegment Point="100,150" />
                        </PathFigure>
                    </PathGeometry>
                </GeometryGroup>
       </Path.Data>
    </Path>
//.....................................................
```

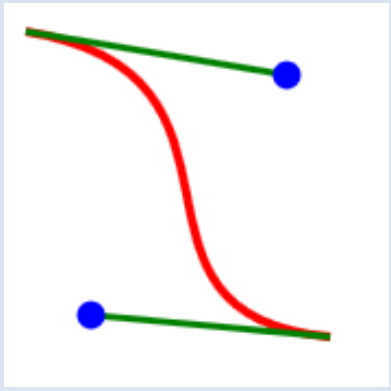
Lab4:

Bezier [Мак-Дональд -370; Петцольд -817]

- `<PathGeometry>` եւ `<PathFigure>` տեգերի միջոցով կարելի է ստանալ ավելի բարդ ֆիգուրաներ:
- `Bezier` գրաֆիկան համարվում է մաթեմեթիկայի ճյուղ: Բանից պարզվում է բեյզերի կորերի միջոցով կարելի է ստանալ ցանկացած տեսքի գծային գրաֆիկա: Չորս կետրի միջոցով կատարվում է կորի ստացում, որտեղ առաջին և վերջին կետերը, դա կորի սկիզբն ու վերջն են: Իսկ միջանկյալ երկու կետերը նշում են այն շոշափողները, որով անցնում է կորը(կարճ ասած այդ միջանկայալ երկու կետերը մագնիսի նման քաշում են կորը իրենց):
- Խնդրի դրվածքն է, ստեղծել բեյզերի կոր, տանել շոշափողներ և մկնիկով շարժել միջանկյալ կետերը:

```
//xaml
<Window x:Class="WpfApplication29.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525" MouseDown="Window_MouseDown"
                           MouseMove="Window_MouseMove" MouseUp="Window_MouseUp">
    <Grid>
        <Path Stroke="Red" StrokeThickness="4">
            <Path.Data>
                <PathGeometry>
                    <PathFigure StartPoint="10,10">
                        <BezierSegment x:Name="x2" Point1="130,30" Point2="40,140"
                                       Point3="150,150"></BezierSegment>
                    </PathFigure>
                </PathGeometry>
            </Path.Data>
        </Path>
        <Path Stroke="green" StrokeThickness="3">
            <Path.Data>
                <GeometryGroup>
                    <LineGeometry StartPoint="10,10" EndPoint="130,30"></LineGeometry>
                    <LineGeometry x:Name="l2" StartPoint="40,140"
                                  EndPoint="150,150"></LineGeometry>
                </GeometryGroup>
            </Path.Data>
        </Path>
        <Path Stroke="blue" StrokeThickness="13">
            <Path.Data>
                <GeometryGroup>
                    <EllipseGeometry     Center="130,30"></EllipseGeometry>
                    <EllipseGeometry x:Name="e2"  Center="40,140"></EllipseGeometry>
                </GeometryGroup>
            </Path.Data>
        </Path>
    </Grid>
</Window>
//xaml.cs
public partial class MainWindow : Window
    {
        bool b;
        public MainWindow()
        {
            InitializeComponent();
        }
        void Window_MouseDown(object sender, MouseButtonEventArgs e)
        {
          b = true;
        }
```

```
        void Window_MouseMove(object sender, MouseEventArgs e)
        {
            if(b)
            {
                x2.Point2 = new Point(e.GetPosition(this).X, e.GetPosition(this).Y);
                l2.StartPoint = new Point(e.GetPosition(this).X, e.GetPosition(this).Y);
                e2.Center = new Point(e.GetPosition(this).X, e.GetPosition(this).Y);
            }
        }
        //private void Window_MouseMove(object sender, MouseEventArgs e)
        //{
        //  if(b && Math.Abs(x-e2.Center.X)<e2.RadiusX && Math.Abs(e2.Center.Y-y)<e2.RadiusY)
        //    {
        //        x = e.GetPosition(this).X;
        //        y = e.GetPosition(this).Y;
        //        {
        //          x2.Point2 = new Point(e.GetPosition(this).X, e.GetPosition(this).Y);
        //          l2.StartPoint = new Point(e.GetPosition(this).X, e.GetPosition(this).Y);
        //          e2.Center = new Point(e.GetPosition(this).X, e.GetPosition(this).Y);
        //        }
        //    }
        //}
        private void Window_MouseUp(object sender, MouseButtonEventArgs e)
        {
               b = false;
        }
    }
```



```
// Bezier Circle
 <Canvas>
        <Path Canvas.Left="150" Canvas.Top="150" Stroke="Black">
            <Path.Data>
                <PathGeometry>
                    <PathGeometry.Figures>
                        <PathFigure StartPoint="0,100">
                            <PolyBezierSegment Points="55 100, 100 55, 100 0,
                                                       100 -55, 55 -100, 0 -100,
                                                       -55 -100, -100 -55, -100 0,
                                                       -100 55, -55 100, 0 100" />
                        </PathFigure>
                    </PathGeometry.Figures>
                </PathGeometry>
            </Path.Data>
        </Path>
    </Canvas>
```

Lab5: [Мак-Дональд –376 ; Натан – 356]

❖ **Play Chess**

- Տեղադրել շախմատի դաշտ: Տեղադրել երկու ցանկացած ֆիգուր: Շարժել ֆիգուրաները մկնիկի միջոցով: Ապահովել նրանց վանդակի կենտրոնում: Կիրառել SetZIndex մեթոդը ֆիգուրաների համար:

```
<Window x:Class="WpfApplicationChess.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="438" Width="416" ResizeMode="CanMinimize"
        MouseMove="Window_MouseMove" >
    <Grid>
        <Image Name="MyBoard" Source="board.jpg" Stretch="Fill"/>
        <Image Name="MyBnFigure" Source="bn.gif" Margin="0,0,0,0" VerticalAlignment="Top"
         HorizontalAlignment="Left"  Width="50" Height="50"
         MouseDown="MyBnFigure_MouseDown" MouseUp="MyBnFigure_MouseUp"  Cursor="Hand"/>
        <Image Name="MyWbFigure" Source="wb.gif" Margin="350,0,0,0"
         VerticalAlignment="Top" HorizontalAlignment="Left" Width="50" Height="50"
         MouseDown="MyWbFigure_MouseDown"  MouseUp="MyWbFigure_MouseUp" Cursor="Hand"/>
        <Grid Name="ramka"></Grid>
    </Grid>
</Window>
//cs
public partial class MainWindow : Window
{
    bool BnFigureClicked, WbFigureClicked;
    double DeltaX, DeltaY;
    public MainWindow()
    {        InitializeComponent();         }
    void Window_MouseMove(object sender, MouseEventArgs e)
    {
        if (BnFigureClicked)
            MyBnFigure.Margin = new Thickness(e.GetPosition(this).X - DeltaX,
            e.GetPosition(this).Y - DeltaY, 0, 0);
        if (WbFigureClicked)
            MyWbFigure.Margin = new Thickness(e.GetPosition(this).X - DeltaX,
            e.GetPosition(this).Y - DeltaY, 0, 0);
    }
    void MyBnFigure_MouseDown(object sender, MouseButtonEventArgs e)
    {
        if (e.ButtonState == e.LeftButton)
        {
            StackPanel.SetZIndex(MyBnFigure, 1);
            StackPanel.SetZIndex(MyWbFigure, 0);
            BnFigureClicked = true;
            DeltaX = e.GetPosition(this).X - MyBnFigure.Margin.Left;
            DeltaY = e.GetPosition(this).Y - MyBnFigure.Margin.Top;
    }   }
    void MyBnFigure_MouseUp(object sender, MouseButtonEventArgs e)
    {
        BnFigureClicked = false;
        //MyBnFigure.Margin = new Thickness((int)(MyBnFigure.Margin.Left + 25) / 50 *
        //50,(int)(MyBnFigure.Margin.Top + 25) / 50 * 50, 0, 0);
    }
    void MyWbFigure_MouseDown(object sender, MouseButtonEventArgs e)
    {
        if (e.ButtonState == e.LeftButton)
        {
            StackPanel.SetZIndex(MyWbFigure, 1);
            StackPanel.SetZIndex(MyBnFigure, 0);
```

```
            WbFigureClicked = true;
            DeltaX = e.GetPosition(this).X - MyWbFigure.Margin.Left;
            DeltaY = e.GetPosition(this).Y - MyWbFigure.Margin.Top;
        }
       // Փղի հարվածներ
            for (int i = -8; i < 8; i++)
            {
                Image im = new Image();
                im.Width = 50;
                im.Height = 50;
                im.VerticalAlignment = MyWbFigure.VerticalAlignment;
                im.HorizontalAlignment = MyWbFigure.HorizontalAlignment;
                im.Source = new BitmapImage(new Uri("ramka.gif", UriKind.Relative));
                ramka.Children.Add(im);
                im.Margin = new Thickness((int)(MyWbFigure.Margin.Left - 50 * i) / 50 * 50,
                                  (int)(MyWbFigure.Margin.Top+50 * i ) / 50 * 50, 0, 0);
            }
            for (int i = -8; i < 8; i++)
            {
                Image im = new Image();
                im.Width = 50;
                im.Height = 50;
                im.VerticalAlignment = MyWbFigure.VerticalAlignment;
                im.HorizontalAlignment = MyWbFigure.HorizontalAlignment;
                im.Source = new BitmapImage(new Uri("ramka.gif", UriKind.Relative));
                ramka.Children.Add(im);
                im.Margin = new Thickness((int)(MyWbFigure.Margin.Left + 50 * i) / 50 * 50,
                                  (int)(MyWbFigure.Margin.Top + 50 * i) / 50 * 50, 0, 0);
            }
    }
    void MyWbFigure_MouseUp(object sender, MouseButtonEventArgs e)
    {
        WbFigureClicked = false;
        MyWbFigure.Margin = new Thickness((int)(MyWbFigure.Margin.Left + 25) / 50 * 50,
        (int)(MyWbFigure.Margin.Top + 25) / 50 * 50, 0, 0);
        ramka.Children.Clear();
}   }
```

❖ Images, (Random - ZAR)

- Գոյություն ունեցող զառի նկարներով նետել զառ: Այն կարելի է իրականացնել string -ի մեջ օգտագործելով int տիպի պատահական թիվ:

```
<Grid>
    <Image Name="image1" HorizontalAlignment="Left" Stretch="Fill"  Width="80" Height="80" />
    <Image Name="image2" HorizontalAlignment="Right" Stretch="Fill" Width="80" Height="80" />
    <Button Content="ԶԱՌ" Name="button1" Height="23" Width="75" Click="button1_Click" />
</Grid>
//cs
    public partial class MainWindow : Window
    {
        public MainWindow()
        {            InitializeComponent();          }
        void button1_Click(object sender, RoutedEventArgs e)
        {   Random rd=new Random();
            int a1 = rd.Next(1,7);
            int a2 = rd.Next(1,7);              // Uniform Resource Identifier (URI)
            image1.Source = new BitmapImage(new Uri("C:\\zar\\zar" + a1 + ".gif"));
            image2.Source = new BitmapImage(new Uri("zar" + a2 + ".gif", UriKind.Relative));
    }   }
```

Lab6:

Menu [Мак-Дональд 780; Натан - 298]

Dialog [Мак-Дональд -696]

- Մենյուի եւ ենթամենյուի օգնությամբ կանչել դիալոգային պատուհան: Դիալոգային պատուհանը կարող է լինել մոդալ կամ ոչ մոդալ՝ ShowDialog(), Show(): Մոդալությունը որոշում է, թե պատուհանը արգելում է, կամ չի արգելում այլ պատուհանների աշխատանքը:
- Անհրաժեշտ է նոր պատուհանում մուտքագրել տեքստ, որի պատասխանը կարտապատկերվի գլխավոր պատուհանի տեքստային դաշտում:

```
//xaml
<Grid>
    <Menu IsMainMenu="True" Height="26" VerticalAlignment="Top">
    <MenuItem Header="File">
            <MenuItem Header="new..." Click="MenuItem_Click">
                </MenuItem>
            <MenuItem Header="copy..."/>
    </MenuItem>
    </Menu>
    <TextBox Name="nnn" Width="200" Height="100" Margin="0,25,0,0"
      VerticalAlignment="Top" ScrollViewer.VerticalScrollBarVisibility="Visible"/>
    <ListBox Name="nnn1" Height="100" Width="200" HorizontalAlignment="Left"
     Margin="45,125,0,0" VerticalAlignment="Bottom" ScrollViewer.CanContentScroll="False"/>
</Grid>
//cs
public partial class MainWindow : Window
{
    public Window w;
    public TextBox tb;
    public MainWindow()
    {
        InitializeComponent();
    }
    void MenuItem_Click(object sender, RoutedEventArgs e)
    {
        w = new Window();
        tb = new TextBox();
        tb.Width = 400;
        tb.Height = 100;
        tb.TextWrapping = TextWrapping.Wrap;
        tb.AcceptsReturn = true;
        Button btn = new Button();
        btn.Width = 150;
        btn.Height = 50;
        btn.VerticalAlignment = VerticalAlignment.Bottom;
        btn.Click += Btn_Click;
        Grid grid = new Grid();
        grid.Children.Add(tb);
        grid.Children.Add(btn);
        w.Content = grid;
        w.ShowDialog();//w.Show();
    }
    void Btn_Click (object sender, RoutedEventArgs e)
    {
            nnn.Text = tb.Text;
            nnn1.Items.Add(tb.Text);
    }
```

❖ Տնային Word –ի Find “text” դիալոգային պատուհանով իրականացում

Lab7:

File Read/Write [Nagel(2012)-673]

- Օրինակ՝ կատարել տեքստի մուտք\ելք ֆայլում: Դրա համար անհրաժեշտ է նշել ֆայլի անունը, հավաքել տեքստը եւ այնուհետեւ Menu -ի միջոցով ապահովել տեքստի գրանցումը եւ կարդալը:


```
//xaml
<Grid>
        <Menu IsMainMenu="True" Height="26" VerticalAlignment="Top">
            <MenuItem Name="write" Header="Write" Click="write_Click" />
            <MenuItem Name="read" Header="Read" Click="read_Click"/>
        </Menu>
        <Label Content="File Name" Margin="90,140,0,0"></Label>
        <TextBox Name="fn" Height="30"  Width="250"  HorizontalAlignment="Left"
                 TextWrapping="Wrap" Background="Bisque"  />
        <Label Content="Text  Input and Write File" Margin="320,140,0,0"></Label>
        <TextBox Name="txtBox" Height="60" Width="250" TextWrapping="Wrap"
                 AcceptsReturn="True" Background="AliceBlue" />
        <Label Content="Text Read from File" Margin="600,140,0,0"></Label>
        <TextBlock Name="txtBlock" Height="60" Width="250" Background="Aqua"
                 HorizontalAlignment="Right" TextWrapping="Wrap"/>
</Grid>

//cs
using System.IO;
void write_Click(object sender, RoutedEventArgs e)
{
      File.WriteAllText(fn.Text, txtBox.Text);
}
void read_Click(object sender, RoutedEventArgs e)
{
      txtBlock.Text = File.ReadAllText(fn.Text);
}
```

❖ **ToolBar (image, copy, cut, past, undo, redo, delete)**

[Мак-Дональд -786; Натан – 351]

- **TollBar** -ը իրենից ներկայացնում է “կոնտեյներ”, որտեղ կարելի է տեղակայել ղեկավարման էլեմենտներ, պատկերներ եւ այլն:
- Օրինակում անհրաժեշտ է իրականացնել ստանդարտ փաթեթների **copy, cut, past, undo, redo**, հնարավորությունները, որոնք աշխարում են տեքստային դաշտի հետ: Իրականացնել տեքստի գրանցում ֆայլում եւ կարդալ այնտեղից: Տեքստի նշումը կատարվում է մկնիկով:

```
//xaml
<Grid>
  <ToolBar Name="tool" Height="50" VerticalAlignment="Top" Background="Azure">
  <Image Source="C:\1\wq.gif"></Image>
  <Button Name="b_copy" Content="copy" ToolTip="katarel copia"  Height="40" Width="30"
               Click="b_copy_Click" ></Button>
  <Button Name="b_cut" Content="cut" Height="40" Width="30" Click="b_cut_Click"></Button>
  <Button Name="b_past" Content="past" Height="40" Width="30" Click="b_past_Click"></Button>
  <Button Name="b_undo" Content="undo" Height="40" Width="30" Click="b_undo_Click"></Button>
  <Button Name="b_redo" Content="redo" Height="40" Width="30" Click="b_redo_Click"></Button>
  </ToolBar>
  <TextBox Name="t" Width="150" Height="150" TextWrapping="Wrap" AcceptsReturn="True"
   Background="Azure" ScrollViewer.VerticalScrollBarVisibility="Visible"></TextBox>
</Grid>
//cs
public partial class MainWindow : Window
{
    string s;
    public MainWindow()
    {
        InitializeComponent();
    }
    void b_copy_Click(object sender, RoutedEventArgs e)
    {
        s = t.SelectedText;
        b_copy.Background = new SolidColorBrush(Colors.Bisque);
    }
    void b_cut_Click(object sender, RoutedEventArgs e)
    {
        s = t.SelectedText;
        t.SelectedText = "";
        b_cut.Background = new SolidColorBrush(Colors.Bisque);
    }
    void b_past_Click(object sender, RoutedEventArgs e)
    {
        t.SelectedText = s;
        b_past.Background = new SolidColorBrush(Colors.Bisque);
    }
    void b_undo_Click(object sender, RoutedEventArgs e)
    {
        t.Undo();
        b_undo.Background = new SolidColorBrush(Colors.Bisque);
    }
    void b_redo_Click(object sender, RoutedEventArgs e)
    {
        t.Redo();
        b_redo.Background = new SolidColorBrush(Colors.Bisque);
    }
}
```

.....

Lab8:

TreeView (ծառի ստեղծում) [Мак-Дональд -672; Натан – 349]

- `TreeView` –ն հնարավորություն է տալիս "folder explorer" -ի նման ներկայացնել ինֆորմացիայի ընտրություն եւ մենյուի նման կատարման գործողություն:
- Օրինակը ցուցաբերում է, թէ ինչպես `Grid` -ի մեկ սյունում ընտրել համակարգչի սարքերը, կամ նրա ենթակա "ֆոլդերները", իսկ մյուսում ցուցադրել ֆայլերը:

```
//xaml //explorer
<Grid>
     <Grid.ColumnDefinitions>
        <ColumnDefinition></ColumnDefinition>
        <ColumnDefinition></ColumnDefinition>
     </Grid.ColumnDefinitions>
      <TreeView Name="W1" Grid.Column="0"
      TreeViewItem.Expanded="DirectoryTreeView_Expanded"
      TreeViewItem.Selected="DirectoryTreeView_Selected"></TreeView>
       <TreeView Name="W2" Grid.Column="1"></TreeView>
</Grid>

//xaml.cs
using System.IO;
public partial class MainWindow : Window
{
      public MainWindow()
      {
            InitializeComponent();
            foreach (DriveInfo drive in DriveInfo.GetDrives())
            {
                  TreeViewItem item = new TreeViewItem();
                  item.Tag = drive;
                  item.Header = drive.ToString();
                  item.Items.Add("*"); //item.Items.Add(drive);
                  W1.Items.Add(item);
            }
      }
      void DirectoryTreeView_Selected(object sender, RoutedEventArgs e)
      {
             TreeViewItem item = (TreeViewItem)e.OriginalSource;
             DirectoryInfo dir;
             W2.Items.Clear();
             if (item.Tag is DriveInfo)
             {
                   DriveInfo drive = (DriveInfo)item.Tag;
                   dir = drive.RootDirectory;
             }
             else
             {
                  dir = (DirectoryInfo)item.Tag;
             }
             foreach (FileInfo file in dir.GetFiles())
             {
                   TreeViewItem newItem = new TreeViewItem();
                   newItem.Tag = file;
                   newItem.Header = file.ToString();
                   W2.Items.Add(newItem);
             }
     }
```

```
    void DirectoryTreeView_Expanded(object sender, RoutedEventArgs e)
    {
        TreeViewItem item = (TreeViewItem)e.OriginalSource;
        item.Items.Clear();
        DirectoryInfo dir;
        if (item.Tag is DriveInfo)
        {
            DriveInfo drive = (DriveInfo)item.Tag;
            dir = drive.RootDirectory;
        }
        else
            dir = (DirectoryInfo)item.Tag;
        foreach (DirectoryInfo d in dir.GetDirectories())
        {
            TreeViewItem newItem = new TreeViewItem();
            newItem.Tag = d;
            newItem.Header = d.ToString();
            newItem.Items.Add("*");
            item.Items.Add(newItem);
        }
    }
}
```

- Լսարանի խնդիր՝ աջ մասում նույնպես ցուցադրվեն ինչպես ֆայլեր այնպես էլ ֆոլդրներ:
- Տնային՝ աջ մասում ֆոլդրները նույնպես հնարավոր լինի բացել իրենց պարունակությամբ:

.....

Lab9:

Sound + Video [Мак-Дональд -805 ; Натан – 722]

- `MediaPlayer` կլասի միջոցով ստեղծել երաժշտության ֆայլի կատարում, որտեղ հնարավոր լինի ձայնի ղեկավարում `Slider` -ով, կատարման տեղի փոփոխություն `Slider` -ով, ինչպես նաեւ `Button` -ով Play, Stop, Pause կազմակերպում:
- `MediaElement` կլասի միջոցով ստեղծել վիդեո ցուցադրում և `Slider` -ով կատարման տեղի փոփոխություն:

- **Audio**

```
//xaml
<Grid>
    <Slider Name="seek" Height="30" ValueChanged="seek_ValueChanged" Value="0" Minimum="0"
            Maximum="100"></Slider>
    <Slider Name="vol" Height="30" ValueChanged="vol_ValueChanged" VerticalAlignment="Top"
            Minimum="0" Value="0.5" Maximum="1"></Slider>
    <Button Name="play" Click="Click_play" Height="30" Width="80" HorizontalAlignment="Left"
            VerticalAlignment="Bottom"> play</Button>
    <Button Name="stop" Click="Click_stop" Height="30" Width="80" HorizontalAlignment="Right"
            VerticalAlignment="Bottom">stop</Button>
    <Button Name="pause" Click="Click_pause" Height="30" Width="80"
            VerticalAlignment="Bottom">pause</Button>
</Grid>
//cs
public partial class MainWindow : Window
{
    MediaPlayer mp = new MediaPlayer();
    public MainWindow()
    {
        InitializeComponent();
        mp.Open(new Uri("C:\\1\\sd.mp3", UriKind.Relative));
    }
```

```
    void Click_play(object sender, RoutedEventArgs e)
    {
        mp.Play();
        // seek.Value = mp.Position.TotalSeconds;
    }
    void Click_stop(object sender, RoutedEventArgs e)
    {
        mp.Stop();
        seek.Value = 0;
    }
    void Click_pause(object sender, RoutedEventArgs e)
    {
        mp.Pause();
    }
    void seek_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)
    {
        seek.Maximum = mp.NaturalDuration.TimeSpan.TotalSeconds; // մարկերի ճիշտ տեղաշարժ
        mp.Position = TimeSpan.FromSeconds(seek.Value);
    }
    void vol_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)
    {
        mp.Volume = vol.Value;
    }
}
```

❖ **Video play**

```
//xaml
<Grid>
        <MediaElement Name="me" Width="400" LoadedBehavior="Manual" Source="C:/1/vi.mp4"/>
        <Button Height="30" Width="60" Click="button_Click"
                HorizontalAlignment="Left">video</Button>
        <Slider Name="seek"  Height="20" Maximum="50"  Minimum="0" Value="0"
                 VerticalAlignment="Bottom" ValueChanged="seek_ValueChanged"/>
</Grid>
//cs
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }
        void button_Click(object sender, RoutedEventArgs e)
        {
            me.Position = TimeSpan.Zero;
            me.Play();
        }
        void seek_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)
        {
            //me.Position = TimeSpan.FromSeconds(e.NewValue);
            me.Position = TimeSpan.FromSeconds(seek.Value);
            me.Play();
        }
    }
```

❖ **Video play with marker**

```
//xaml
<Grid>
        <MediaElement Name="me" Width="400" LoadedBehavior="Manual"
                Source="C:/1/vi.mp4" MediaOpened="me_MediaOpened"/>
        <Button Height="30" Width="60" Click="button_Click" HorizontalAlignment="Left">
         Play</Button>
        <Slider Name="seek" Height="20" VerticalAlignment="Bottom"
        Maximum="50" ValueChanged="seek_ValueChanged"/>
</Grid>
//cs
using System.Windows.Threading;
public partial class MainWindow : Window
{
        DispatcherTimer timer;
        public MainWindow()
        {
            InitializeComponent();
            timer= new DispatcherTimer();
            timer.Tick += Timer_Tick;
            //timer.Interval = TimeSpan.FromMilliseconds(10);
        }
        void Timer_Tick(object sender, EventArgs e)
        {
            seek.Value = me.Position.TotalMilliseconds;
        }
        void button_Click(object sender, RoutedEventArgs e)
        {
            //me.Position = TimeSpan.Zero;
            me.Play();
        }
        void seek_ValueChanged(object sender,RoutedPropertyChangedEventArgs<double> e)
        {
            me.Position = TimeSpan.FromMilliseconds(seek.Value);
            me.Play();
        }
        void me_MediaOpened(object sender, RoutedEventArgs e)
        {
            seek.Maximum = me.NaturalDuration.TimeSpan.TotalMilliseconds;
            timer.Start();
        }
}
```

Lab10:

Drawing 3D

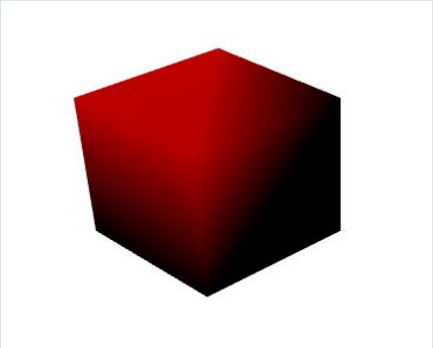
[Мак-Дональд-827; Натан -602]

- WPF -ում նրդրված է 3D գրաֆիկական համակարգ: Ինչպես գիտենք 3D գրաֆիկա կարելի է ստանալ նաեւ 2D համակարգով, ցուցադրելով որպես պատկեր: Սակայն, երբ անհրաժեշտ է պակերը ձևափոխել եռաչափ տարածքում, ապա անհրաժեշտ է բարդ մաթեմատիկա եւ ծրագրային ռեսուրս:
- WPF -ի տեգերում եւ նրանց հատկանիշներում ներդրված է 3D հնարավորություն, որը գրաֆիկան դարձնում է ավելի արդյունավետ:
- 3D գրաֆիկայի համար անհրաժեշտ է առաջին՝ պատկերի կառուցում: Երկրորդ՝ դիտակետի կամերայի տեղադրում եւ երրորդ՝ պատկերի լուսավորում: Նշված երեք ծրագրային մասերը տեղադրվում են <Viewport3D> տեգում:
- WPF –ի 3D գրաֆիկական իրականացման միավոր է հանդիսանում եռանկյունը եւ պատկերները ստացվում են եռանկյան բազմության միջոցով:
  - Առաջին օրինակը ցուցադրում է եռանկյուն:
  - Երկրորդ օրինակը ցուցադրում է խորանարդ: Նշենք, որ խորանարդը ստանալու համար նրա նիստերը բաժանվում են եռանկյունների, որոնց քանակն է 12: Խորանարդի գագաթային կետերը, որոնք ծառայում են որպես եռանկյան գագաթներ, 8 -ն են: Օրինակում կետերի քանակը ներկայցվում է 24 ով, որպեսզի խորանարդի եզրերը լինեն ընդգծված:

❖ Եռանկյան օրինակ

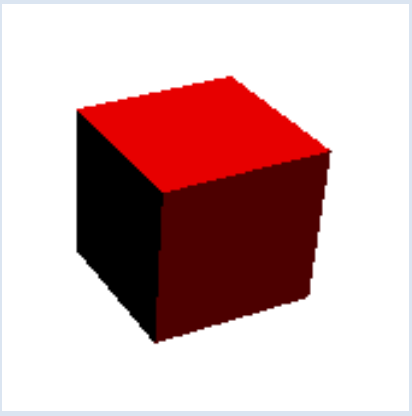
```
//xaml
<Grid>
    <Viewport3D>
        <Viewport3D.Camera>
        <PerspectiveCamera Position="-3,3,3" LookDirection="2,-2,-2" UpDirection="0,1,0"/>
        </Viewport3D.Camera>
        <ModelVisual3D>
            <ModelVisual3D.Content>
                <DirectionalLight Color="White" Direction="0,0,-1" />
            </ModelVisual3D.Content>
        </ModelVisual3D>
        <ModelVisual3D>
            <ModelVisual3D.Content>
              <GeometryModel3D>
              <GeometryModel3D.Geometry>
                <MeshGeometry3D Positions="-1,0,0,0,1,0,1,0,0" TriangleIndices="0,2,1" />
              </GeometryModel3D.Geometry>
              <GeometryModel3D.Material>
                 <DiffuseMaterial Brush="Yellow" />
              </GeometryModel3D.Material>
              </GeometryModel3D>
            </ModelVisual3D.Content>
        </ModelVisual3D>
    </Viewport3D>
</Grid>
//-------------------------------------------------
```

❖ Խորանարդի օրինակ 8 գագաթով



```
//xaml
<Grid>
   <Viewport3D>
     <Viewport3D.Camera>
     <PerspectiveCamera Position="-40,40,40" LookDirection="40,-40,-40" UpDirection="0,0,1"/>
     </Viewport3D.Camera>
        <ModelVisual3D>
            <ModelVisual3D.Content>
                <Model3DGroup>
                    <DirectionalLight Color="White" Direction="-1,-1,-3" />
                    <GeometryModel3D>
                        <GeometryModel3D.Geometry>
                             <MeshGeometry3D Positions="0,0,0  10,0,0  0,10,0  10,10,0
                                                        0,0,10 10,0,10 0,10,10 10,10,10"
                                        TriangleIndices="0,2,1  1,2,3   0,4,2   2,4,6
                                                         0,1,4  1,5,4   1,7,5   1,3,7
                                                         4,5,6  7,6,5   2,6,3   3,6,7"/>
                        </GeometryModel3D.Geometry>
                        <GeometryModel3D.Material>
                            <DiffuseMaterial Brush="Red"/>
                        </GeometryModel3D.Material>
                    </GeometryModel3D>
                </Model3DGroup>
            </ModelVisual3D.Content>
        </ModelVisual3D>
    </Viewport3D>
</Grid>
```

❖ Խորանարդ 24 գագաթով



❖ Խնդիր: Պտկերի ցուցադրում 3D եռանկյան և խորանարդի նիստերի վրա: [Мак-Дональд]

Lab11:

Անիմացիա [Мак-Дональд -402; Натан - 675]

- WPF –ում անիմացիա կատարելու համար չկա “թայմերի” օգտագործման եւ իրադարձությունների մշակման անհարաժեշտություն: Անիմացիան ֆորմում կատարվում է թայմերի միջոցով անընդհատ վերանկարումով եւ անիմացիայի ընթացքը չի գտնվում ծրագրի հետ ակտիվ կապի մեջ: WPF – ի դեպքում անիմացիան կատարվում է հատկանիշի (property based) փոփոխման միջոցով տրված ժամանակի միջակայքով եւ նա ակտիվ կապի մեջ է ծրագրի հետ:
- Անիմացիոն կլասների երեք խումբ կա `Using Windows.Media.Animation` անունի տարածքում.

✓ Ինտերպոլացիայով - `NameClassAnimation`
✓ Կադրով - `NameClassAnimationUsingKeyFrame`
✓ Path – ով՝ `NameClassAnimationUsingPath`

- Սկզբից ցուցադրենք, Form ով անիմացիա, կատարելով `Button` -ի չափերի փոփոխման միջոցով, միաժամանակ կատարելով մկնիկի “քլիք”: Օրինակը ցույց կտա, որ մկնիկի “քլիք” հնարավոր չէ մինչև անիմացիայի ավարտը:
- Հաջորդ օրինակը կիրականացնենք WPF -ով, որտեղ անիմացիոն հնարավորությունները ընդգրկված են տարբեր էլեմենտների հատկանիշների եւ մեթոդների միջոցով: Կտեսնենք, որ անիմացիայի ժամանակ մկնիկի “քլիք” թույլատրվում է:

❖ **Form Animation** (Button)

```
using System;
using System.Drawing;
using System.Windows.Forms;
class CCC : Form
{
    Button bt, bt2;
    public CCC()
    {
        bt = new Button();
        bt.Parent = this;
        bt2 = new Button();
        bt2.Parent = this;
        bt2.Location = new Point(100, 100);
        bt2.Size = new Size(50, 50);
        bt2.BackColor = Color.Bisque;
        bt.Click += new EventHandler(bt_Click);
        bt2.Click += new EventHandler(bt2_Click);
    }
    void bt2_Click(object sender, EventArgs e)
    {
        MessageBox.Show("barev");
    }
    void bt_Click(object sender, EventArgs e)
    {
        for (int i = 0; i < 100; i++)
        {
            bt2.Width++;
            bt2.Update();
            System.Threading.Thread.Sleep(50);
        }
        bt2.Size = new Size(50, 50);
   }
}
```

❖ **WPF - Property-Based Animation** (Button)

```
//xaml
<Grid>
        <Button Click="Button_Click" Margin="0,0,440,260">start</Button>
        <Button Name="b" Width="100" Height="100" Click="b_Click"></Button>
</Grid>
//xaml.cs
using System.Windows.Media.Animation;
public partial class MainWindow : Window
{
        DoubleAnimation a;
        public MainWindow()
        {
            InitializeComponent();
        }
        void Button_Click(object sender, RoutedEventArgs e)
        {
            a = new DoubleAnimation();
            a.From = 50;
            a.To = 300;
            a.Duration = TimeSpan.FromSeconds(10);
            // a.RepeatBehavior = new RepeatBehavior(2);
            a.RepeatBehavior = RepeatBehavior.Forever;
            b.BeginAnimation(Button.WidthProperty, a);
        }
        void b_Click(object sender, RoutedEventArgs e)
        {
            MessageBox.Show("barev");
        }
}
```

❖ Խնդիր: Կատարել գույնի անիմացիա օգտվելով գրականությունից: [Мак-Дональд]

.....

Lab12:

3D Անիմացիա [Мак-Дональд -852]

❖ **Եռանկյան տարածքային փոփոխություն:**


```
//xaml
<Grid>
    <Viewport3D>
      <Viewport3D.Camera>
         <PerspectiveCamera Position="-2,2,2" LookDirection="2,-2,-2" UpDirection="0,1,0" />
      </Viewport3D.Camera>
        <ModelVisual3D>
            <ModelVisual3D.Content>
                <DirectionalLight Color="White" Direction="0,0,-1" />
            </ModelVisual3D.Content>
```

```
        </ModelVisual3D>
        <ModelVisual3D>
            <ModelVisual3D.Content>
                <GeometryModel3D>
                <GeometryModel3D.Geometry>
                <MeshGeometry3D Positions="-1,0,0,0,1,0,1,0,0" TriangleIndices="0,2,1" />
                </GeometryModel3D.Geometry>
                    <GeometryModel3D.Material>
                        <DiffuseMaterial Brush="Yellow" />
                    </GeometryModel3D.Material>
                </GeometryModel3D>
            </ModelVisual3D.Content>
            <ModelVisual3D.Transform>
                <RotateTransform3D>
                    <RotateTransform3D.Rotation>
                        <AxisAngleRotation3D x:Name="rotate" Axis="0 1 0" />
                    </RotateTransform3D.Rotation>
                </RotateTransform3D>
            </ModelVisual3D.Transform>
        </ModelVisual3D>
    </Viewport3D>
    <Slider Minimum="-100" Maximum="0" Height="22" Margin="66,88,88,77" Name="slider1"
                VerticalAlignment="Top" Value="{Binding ElementName=rotate, Path=Angle}" />
</Grid>
```

- **Խորանարդի անիմացիա** 24 գագաթով

```
//xaml
<Grid>
   <Viewport3D>
     <Viewport3D.Camera>
     <PerspectiveCamera Position="-40,40,40" LookDirection="40,-40,-40" UpDirection="0,0,1"/>
     </Viewport3D.Camera>
        <ModelVisual3D>
            <ModelVisual3D.Content>
                <Model3DGroup>
                    <DirectionalLight Color="White" Direction="-1,-1,-3" />
                    <GeometryModel3D>
                        <GeometryModel3D.Geometry>
                            <MeshGeometry3D Positions="0,0,0  10,0,0   0,10,0  10,10,0
                                                       0,0,0  0,0,10   0,10,0  0,10,10
                                                       0,0,0  10,0,0   0,0,10  10,0,10
                                                       10,0,0 10,10,10 10,0,10 10,10,0
                                                       0,0,10 10,0,10  0,10,10 10,10,10
                                                       0,10,0 0,10,10  10,10,0 10,10,10"
                                        TriangleIndices="0,2,1    1,2,3
                                                         4,5,6    6,5,7
                                                         8,9,10   9,11,10
                                                         12,13,14 12,15,13
                                                         16,17,18 19,18,17
                                                         20,21,22 22,21,23 " />
                        </GeometryModel3D.Geometry>
                        <GeometryModel3D.Material>
                            <DiffuseMaterial Brush="Red"/>
                        </GeometryModel3D.Material>
                        <GeometryModel3D.Transform>
                            <TranslateTransform3D x:Name="myTranslateTransform3D"
                                          OffsetX="0" OffsetY="0" OffsetZ="0" />
                        </GeometryModel3D.Transform>
                    </GeometryModel3D>
```

```
                </Model3DGroup>
            </ModelVisual3D.Content>
            <ModelVisual3D.Transform>
                <RotateTransform3D>
                    <RotateTransform3D.Rotation>
                    <AxisAngleRotation3D x:Name="myAngleRotation" Axis="0 0 1" Angle="12" />
                    </RotateTransform3D.Rotation>
                </RotateTransform3D>
            </ModelVisual3D.Transform>
        </ModelVisual3D>
        <Viewport3D.Triggers>
            <EventTrigger RoutedEvent="Viewport3D.Loaded">
                <BeginStoryboard>
                    <Storyboard>
                        <DoubleAnimation Storyboard.TargetName="myAngleRotation"
                            Storyboard.TargetProperty="Angle"
                            From="0" To="360" Duration="0:0:4" RepeatBehavior="Forever"/>
                    </Storyboard>
                </BeginStoryboard>
            </EventTrigger>
        </Viewport3D.Triggers>
    </Viewport3D>
</Grid>
```

……...…

WPF (պրակտիկ): Ներկայացնում ենք պահանջներ, որնցից կօգտվեն քննական խնդիրները:

1. Layout (Grid, StackPanel, WrapPanel, DockPanel, Canvas) տեգերի կիրառում:
2. Ֆիբոնաչի շարքի (1 1 2 3 5 8 13 21 34 …..) լուծման խնդիր տարբեր ղեկավարման էլեմենտների կիրառումով:
3. Կատալանի շարքի անդմների որոշում ռեկուրսիվ մեթոդով:
4. Ղեկավարման էլեմենտների կիտրառություն [Button, TextBox, ListBox, Label, TexBlock, RadioButton, Slider, ChekBox] XAML տեգերով և C# կոդով:
5. Պատահական Random գույների ստացում և կիրառում
6. Underline , Bold, *Italic* տեքստի կոմբինացիայի կիրառում:
7. Menu, ToolBar կիրառում:
8. Դիալոգային պատուհանի օգտագործում տեքստի փոխանցման ձևով:
9. Դիալոգային պատուհանի միջոցով կատարել տեքստի փնտրում:
10. Մատիտ ծրագրի ստեղծում, չօգտվելով `<InkCanvas>` ից: Փոփոխել մատիտի գույնը և լայնությունը:
11. Line, Rectangle, Ellipse գրաֆիկաներ կլասներով և տեգերով: Մկնիկով չափերի փոփոխաություն:
12. Տարբեր գրաֆիկական ֆիգուրաների միավորում <GeometryGroup> տեգով:
13. Տարբեր գրաֆիկական ֆիգուրաների միավորում <CombinedGeometry> տեգով:
14. Մկնիկով տեղաշարժ <Image> և <Path> տեգերով ներկայացված գրաֆիկաները:
15. Բեզիերի կորի ստացում <BezierSegment> և <PolyBezierSegment> տեգերի միջոցով:

16. Բեզիերի կորի ձևափուխում մկնիկի միջոցով:
17. Շախմատի տախտակի վրա տեղադրել ֆիգուրաներ, ցուցադրել հարվածները:
18. Շախմատի տախտակի վրա տեղադրել ֆիգուրաներ, մկնիկով շարժել ֆիքսելով վանդակների կենտրոններում:
19. Շախմատի տախտակի վրա տեղադրել ֆիգուրաներ, մկնիկով շարժել միայն թույլատրված քայլերով:
20. Տեքստի ֆայլային մուտք/ելքի կազմակերպում ներմուծելով ֆայլի անունը և տեքստը:
21. `TreeView` (explorer) սարքերի և ֆոլդերների ցուցադրում ձախ մասում, ֆայլերի և ֆոլդրների ցուցադրում աջ մասում: Աջ մասում ֆոլդրները նույնպես հնարավոր լինի բացել իրենց պարունակությամբ:
22. MediaPlayer կլասի միջոցով ստեղծել երաժշտության ֆայլի կատարում, որտեղ հնարավոր լինի նշված ղեկավարման էլեմենտներով Play, Stop, Pause և ձայնի ղեկավարում Slider –ով:
23. MediaPlayer կլասի միջոցով ստեղծել երաժշտության ֆայլի կատարում, որտեղ հնարավոր լինի նշված ղեկավարման էլեմենտներով Play և կատարման տեղի փոփոխություն Slider –ով: Մարկերի կատարման տեղի ուղղեկցում:
24. MediaElement կլասի միջոցով ստեղծել վիդեո ցուցադրում նշված ղեկավարման էլեմենտներով Start, Stop, Pause և ձայնի ղեկավարում Slider -ով:
25. MediaElement կլասի միջոցով ստեղծել վիդեո ցուցադրում նշված ղեկավարման էլեմենտներով Start և կատարման տեղի փոփոխություն Slider –ով: Մարկերի կատարման տեղի ուղղեկցում:
26. Կատարել Button ղեկավարման էլեմենտի չափերի և գույնային անիմացիա C# հրամաններով և `MessageBox` միջամտում:
27. Կատարել Button ղեկավարման էլեմենտի չափերի և գույնային անիմացիա XAML տեգերով և `MessageBox` միջամտում:
28. 3D եռանկյան ստացում և անիմացիա նշված առանցքի շուրջ:
29. 3D եռանկյան ստացում և պատկերի ցուցադրում նրա վրա:
30. 3D խորանարդի ստացում և անիմացիա սլայդերի միջոցով:
31. 3D խորանարդի ստացում և պատկերի ցուցադրում նշված նիստի վրա: