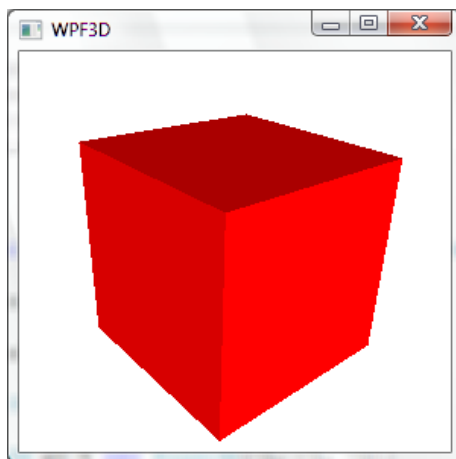


8. WPF - 3D grafika II.

Navážeme na předchozí díl, kde jsme implementovali trojrozměrnou krychli, ze které nyní vytvoříme interaktivní objekt. Co přesně budeme dělat? Vytvoříme video přehrávač, video se bude vykreslovat na přední stranu krychle a na horní stranu krychle vložíme dvě tlačítka pro ovládání videa - *play* a *pause*. Zdá se vám to nemožné? Čtěte dál...

Implementace krychle z minulého dílu



Window1.xaml

```
<Window x:Class="WPF3D.Window1"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:Tools3D="clr-namespace:_3DTools;assembly=3DTools"
  Title="WPF3D" Height="300" Width="300"
  >
  <Tools3D:TrackballDecorator>
  <Viewport3D Name="myViewport3D">
    <Viewport3D.Camera>
      <PerspectiveCamera LookDirection="0,0,-1" Position="0,0,5"
        NearPlaneDistance="3" FarPlaneDistance="100" />
    </Viewport3D.Camera>
    <ModelVisual3D>
      <ModelVisual3D.Content>
        <Model3DGroup>
          <AmbientLight Color="#AA0000" />
          <DirectionalLight Color="White" Direction="0.2,0.5,-1" />
        </Model3DGroup>
      </ModelVisual3D.Content>
    </ModelVisual3D>
  </Viewport3D>
</Tools3D:TrackballDecorator>
</Window>
```

Window1.xaml.cs

```
using System;
using System.Windows;
using System.Windows.Media;
using System.Windows.Media.Media3D;
using System.Windows.Controls;

namespace WPF3D
{
    public partial class Window1 : System.Windows.Window
    {
        public Window1()
        {
            InitializeComponent();

            Model3DGroup models = new Model3DGroup();

            Point3D p0 = new Point3D(-1, -1, -1);
            Point3D p1 = new Point3D(1, -1, -1);
            Point3D p2 = new Point3D(1, -1, 1);
            Point3D p3 = new Point3D(-1, -1, 1);
            Point3D p4 = new Point3D(-1, 1, -1);
            Point3D p5 = new Point3D(1, 1, -1);
            Point3D p6 = new Point3D(1, 1, 1);
            Point3D p7 = new Point3D(-1, 1, 1);

            models.Children.Add(CreateRectangle(p3, p2, p6, p7, Brushes.Red));
            models.Children.Add(CreateRectangle(p2, p1, p5, p6, Brushes.Red));
            models.Children.Add(CreateRectangle(p1, p0, p4, p5, Brushes.Red));
            models.Children.Add(CreateRectangle(p0, p3, p7, p4, Brushes.Red));
            models.Children.Add(CreateRectangle(p7, p6, p5, p4, Brushes.Red));
            models.Children.Add(CreateRectangle(p2, p3, p0, p1, Brushes.Red));

            ModelVisual3D visual = new ModelVisual3D();
            visual.Content = models;

            myViewport3D.Children.Add(visual);
        }

        public GeometryModel3D CreateRectangle(
            Point3D point1, Point3D point2,
            Point3D point3, Point3D point4, Brush brush)
        {
            MeshGeometry3D mesh = new MeshGeometry3D();
            mesh.Positions.Add(point1);
            mesh.Positions.Add(point2);
            mesh.Positions.Add(point3);
            mesh.Positions.Add(point4);

            mesh.TriangleIndices.Add(0);
            mesh.TriangleIndices.Add(1);
            mesh.TriangleIndices.Add(2);

            mesh.TriangleIndices.Add(0);
            mesh.TriangleIndices.Add(2);
            mesh.TriangleIndices.Add(3);

            return new GeometryModel3D(mesh,
                new DiffuseMaterial(brush));
        }
    }
}
```

V projektu jsme použili knihovnu [3D Tools for WPF](#).

Video

Vložení videa do projektu

pravým tlačítkem myši klikněte v *Solution Exploreru* na "WPF3D" projekt > Add > Existing Item > .. a vyberte video soubor

... dále musíme nastavit kopírování souboru:

levým kliknout na video soubor v *Solution Exploreru* > ve vlastnostech (*properties*) nastavit "Copy To Output Directory" na hodnotu "Copy always"

Přehrávání videa

Pro práci s videem máme ve WPF k dispozici třídu *MediaPlayer*

```
MediaPlayer player = new MediaPlayer();
player.Open(new Uri(@"mojeVideo.wmv", UriKind.Relative));
```

Vytvoření textury z videa

Jak vykreslit video na *mesh*? Musíme z videa udělat *Brush* ze kterého můžeme vytvořit texturu.

Co je *Brush*? Jedná se o výplň typu ...

- barva (*SolidColorBrush*)
- lineární gradient (*LinearGradientBrush*)
- radiální gradient (*RadialGradientBrush*)
- obrázek (*ImageBrush*)
- kontroly (*VisualBrush*) - vykreslí se vzhled kontrol
- prvky typu *Drawing* jako jsou text, video ,obrázky a další... (*DrawingBrush*)

Pomocí třídy *VideoDrawing* uděláme z videa *Brush*

```
VideoDrawing vd = new VideoDrawing();
vd.Player = player;
vd.Rect = new Rect(0, 0, 100, 100);
vd.Player.Play();

DrawingBrush db = new DrawingBrush(vd);
Brush videoBrush = (Brush)db;
```

Window1.xaml.cs

```
public partial class Window1 : System.Windows.Window
{
    MediaPlayer mp;
    public Window1()
    {
        InitializeComponent();

        // Video
        mp = new MediaPlayer();
        mp.Open(new Uri(@"mojeVideo.wmv", UriKind.Relative));

        VideoDrawing vd = new VideoDrawing();
        vd.Player = mp;
        vd.Rect = new Rect(0, 0, 100, 100);
        vd.Player.Play();

        DrawingBrush db = new DrawingBrush(vd);
        Brush videoBrush = (Brush)db;

        // Model
        Model3DGroup models = new Model3DGroup();

        Point3D p0 = new Point3D(-1, -1, -1);
        Point3D p1 = new Point3D(1, -1, -1);
        Point3D p2 = new Point3D(1, -1, 1);
        Point3D p3 = new Point3D(-1, -1, 1);
        Point3D p4 = new Point3D(-1, 1, -1);
        Point3D p5 = new Point3D(1, 1, -1);
        Point3D p6 = new Point3D(1, 1, 1);
        Point3D p7 = new Point3D(-1, 1, 1);

        models.Children.Add(CreateRectangle(p3, p2, p6, p7, videoBrush));
        models.Children.Add(CreateRectangle(p2, p1, p5, p6, Brushes.Red));
        models.Children.Add(CreateRectangle(p1, p0, p4, p5, Brushes.Red));
        models.Children.Add(CreateRectangle(p0, p3, p7, p4, Brushes.Red));
        models.Children.Add(CreateRectangle(p7, p6, p5, p4, Brushes.Red));
        models.Children.Add(CreateRectangle(p2, p3, p0, p1, Brushes.Red));

        ModelVisual3D visual = new ModelVisual3D();
        visual.Content = models;
        myViewport3D.Children.Add(visual);
    }

    public GeometryModel3D CreateRectangle(Point3D point1, Point3D point2,
        Point3D point3, Point3D point4, Brush brush)
    {
        MeshGeometry3D mesh = new MeshGeometry3D();
        mesh.Positions.Add(point1);
        mesh.Positions.Add(point2);
        mesh.Positions.Add(point3);
        mesh.Positions.Add(point4);

        mesh.TriangleIndices.Add(0);
        mesh.TriangleIndices.Add(1);
        mesh.TriangleIndices.Add(2);

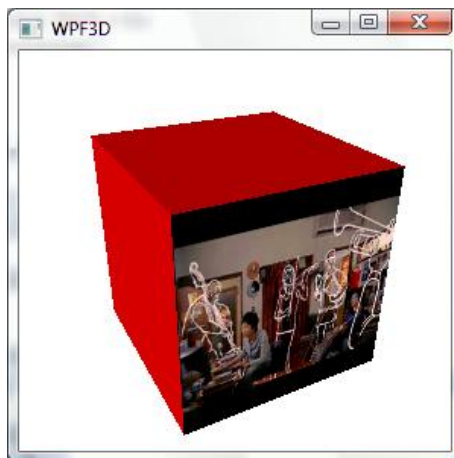
        mesh.TriangleIndices.Add(0);
        mesh.TriangleIndices.Add(2);
        mesh.TriangleIndices.Add(3);

        mesh.TextureCoordinates.Add(new Point(0, 1));
        mesh.TextureCoordinates.Add(new Point(1, 1));
        mesh.TextureCoordinates.Add(new Point(1, 0));
        mesh.TextureCoordinates.Add(new Point(0, 0));

        return new GeometryModel3D(mesh, new DiffuseMaterial(brush)); } }
}
```

1. Abychom mohli pracovat s videem, vytvořili jsme *MediaPlayer*
2. Video jsme zkonvertovali přes *MediaPlayer > VideoDrawing > DrawingBrush > Brush*, abychom ho mohli použít jako texturu
3. Protože nyní jako texturu používáme video, musíme při vytváření *mesh* určit *TextureCoordinates* (pole bodů, podle kterých je textura mapována na *mesh*)

A takto vypadá naše aplikace nyní ...



2D na 3D?

Většinou když vytváříme aplikaci, používáme různé 2D kontroly jako jsou tlačítka, textová pole, labely a další prvky. Co kdybychom ale chtěli zobrazit tyto kontroly v prostoru, jde to? Ano, lze je použít jako texturu, ale bude se jednat pouze o ne-interaktivní zobrazení! ... na tlačítka nepůjde kliknout, do textboxů nepůjde zadávat text, bude se v podstatě jednat pouze o obrázky těchto kontrol. Implicitně tedy nemůžeme ve WPF používat funkční 2D kontroly v prostoru.

Tým, který má na starost 3D grafiku ve WPF, se ale rozhodl poskytnout implementaci, se kterou bude možné 2D prvky na 3D povrchu zobrazit interaktivně! Tato implementace je k dispozici v *3D Tools for WPF*, který používáme v naší aplikaci. Toho využijeme pro ...

Ovládání videa

Vytvoříme dvě tlačítka (*play* a *pause*), kterými budeme ovládat video. Tyto tlačítka budou zobrazeny na vrchní stěně krychle.

Implementace

Pokud chceme použít 2D obsah na 3D mesh, musíme ...

1. Vložit *Viewport3D* do *Interactive3DDecorator* (nachází se v *3D Tools for WPF*)
2. Vytvořit *Interactive3DVisual*
 - do *Interactive3DVisual.Geometry* přiřadit mesh, na který se kontroly vykreslí
 - do *Interactive3DVisual.Visual* vložit kontroly, které se na *mesh* mají vyrenderovat

Window1.xaml

```
<Window x:Class="WPF3D.Window1"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:Tools3D="clr-namespace:_3DTools;assembly=3DTools"
  Title="WPF3D" Height="300" Width="300"
  >
  <Window.Resources>
    <Grid x:Key="VideoControl" Background="Red">
      <Grid.ColumnDefinitions>
        <ColumnDefinition />
        <ColumnDefinition />
      </Grid.ColumnDefinitions>
      <Grid.RowDefinitions>
        <RowDefinition Height="40" />
        <RowDefinition Height="40" />
      </Grid.RowDefinitions>
      <Button Grid.Column="0" Grid.Row="1" Click="Play">Play</Button>
      <Button Grid.Column="1" Grid.Row="1" Click="Pause">Pause</Button>
    </Grid>
  </Window.Resources>
  <Tools3D:TrackballDecorator>
    <Tools3D:Interactive3DDecorator>
      <Viewport3D Name="myViewport3D">
        <Viewport3D.Camera>
          <PerspectiveCamera LookDirection="0,0,-1" Position="0,0,5"
            NearPlaneDistance="3" FarPlaneDistance="100" />
        </Viewport3D.Camera>
        <ModelVisual3D>
          <ModelVisual3D.Content>
            <Model3DGroup>
              <AmbientLight Color="#AA0000" />
              <DirectionalLight Color="White" Direction="0.2,0.5,-1" />
            </Model3DGroup>
          </ModelVisual3D.Content>
        </ModelVisual3D>
      </Viewport3D>
    </Tools3D:Interactive3DDecorator>
  </Tools3D:TrackballDecorator>
</Window>
```

Window1.xaml.cs

```
using System;
using System.Windows;
using System.Windows.Media;
using System.Windows.Media.Media3D;
using System.Windows.Controls;
using _3DTools;

namespace WPF3D
{
    public partial class Window1 : System.Windows.Window
    {
        MediaPlayer mp;
        bool paused = false;

        public Window1()
        {
            InitializeComponent();

            // Video
            mp = new MediaPlayer();
            mp.Open(new Uri(@"mojeVideo.wmv", UriKind.Relative));

            VideoDrawing vd = new VideoDrawing();
            vd.Player = mp;
            vd.Rect = new Rect(0, 0, 100, 100);
            vd.Player.Play();

            DrawingBrush db = new DrawingBrush(vd);
            Brush videoBrush = (Brush)db;

            // Model
            Model3DGroup models = new Model3DGroup();

            Point3D p0 = new Point3D(-1, -1, -1);
            Point3D p1 = new Point3D(1, -1, -1);
            Point3D p2 = new Point3D(1, -1, 1);
            Point3D p3 = new Point3D(-1, -1, 1);
            Point3D p4 = new Point3D(-1, 1, -1);
            Point3D p5 = new Point3D(1, 1, -1);
            Point3D p6 = new Point3D(1, 1, 1);
            Point3D p7 = new Point3D(-1, 1, 1);

            models.Children.Add(
                CreateRectangle(p3, p2, p6, p7, videoBrush)); // predni
            models.Children.Add(
                CreateRectangle(p2, p1, p5, p6, Brushes.Red)); // pravy
            models.Children.Add(
                CreateRectangle(p1, p0, p4, p5, Brushes.Red)); // zadni
            models.Children.Add(
                CreateRectangle(p0, p3, p7, p4, Brushes.Red)); // levy
            models.Children.Add(
                CreateRectangle(p2, p3, p0, p1, Brushes.Red)); // spodni

            // vrchni
            MeshGeometry3D interactiveMesh;
            CreateRectangle(p7, p6, p5, p4, Brushes.Red, out interactiveMesh);

            InteractiveVisual3D interactive = new InteractiveVisual3D();
            interactive.Geometry = interactiveMesh;
        }
    }
}
```

```
interactive.Visual = (Visual)FindResource("VideoControl");
myViewport3D.Children.Add(interactive);
```

```
ModelVisual3D visual = new ModelVisual3D();
visual.Content = models;
myViewport3D.Children.Add(visual);
}
```

```
void Play(object sender, RoutedEventArgs e)
{
    mp.Stop();
    mp.Play();
    paused = false;
}
```

```
void Pause(object sender, RoutedEventArgs e)
{
    if (paused) { mp.Play(); paused = false; }
    else        { mp.Pause(); paused = true; }
}
```

```
public GeometryModel3D CreateRectangle(
    Point3D point1, Point3D point2,
    Point3D point3, Point3D point4,
    Brush brush)
{
    MeshGeometry3D mesh;
    return CreateRectangle(point1,point2,point3,point4,brush,out mesh);
}
```

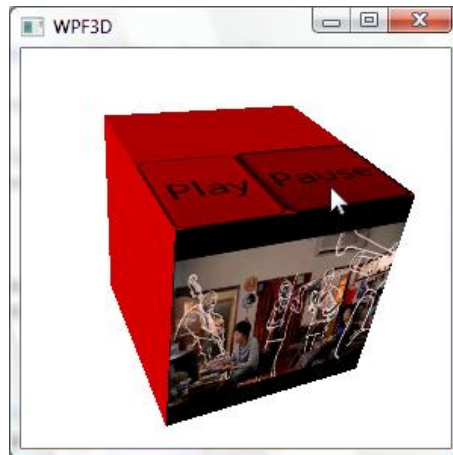
```
public GeometryModel3D CreateRectangle(
    Point3D point1, Point3D point2,
    Point3D point3, Point3D point4,
    Brush brush, out MeshGeometry3D mesh)
{
    mesh = new MeshGeometry3D();
    mesh.Positions.Add(point1);
    mesh.Positions.Add(point2);
    mesh.Positions.Add(point3);
    mesh.Positions.Add(point4);

    mesh.TriangleIndices.Add(0);
    mesh.TriangleIndices.Add(1);
    mesh.TriangleIndices.Add(2);

    mesh.TriangleIndices.Add(0);
    mesh.TriangleIndices.Add(2);
    mesh.TriangleIndices.Add(3);

    mesh.TextureCoordinates.Add(new Point(0, 1));
    mesh.TextureCoordinates.Add(new Point(1, 1));
    mesh.TextureCoordinates.Add(new Point(1, 0));
    mesh.TextureCoordinates.Add(new Point(0, 0));

    return new GeometryModel3D(mesh,
        new DiffuseMaterial(brush));
}
}
```



Závěr

3D ve WPF není určeno pro vývoj složitých 3D scén nebo her, slouží spíše pro vytváření trojrozměrných efektů nebo jednoduchých aplikací. Pro pokročilejší práci s trojrozměrnou grafikou máme k dispozici Direct3D, které pokud potřebujeme můžeme ve WPF také použít.

Více informací o WPF naleznete na www.unitedstatesof.net/wpf

Aleš Šturala, 18. 2. 2007

<http://hidentity.org/hid/CZ123456>