В.В. Подбельский

Использованы иллюстрации пособия Daniel Solis, Illustrated C#

Иллюстрации к курсу лекций по дисциплине «Программирование на C#»

12. Часть 3

Свойства в XAML и привязка данных. Аналоговые часы

Приоритеты для службы зависимых свойств

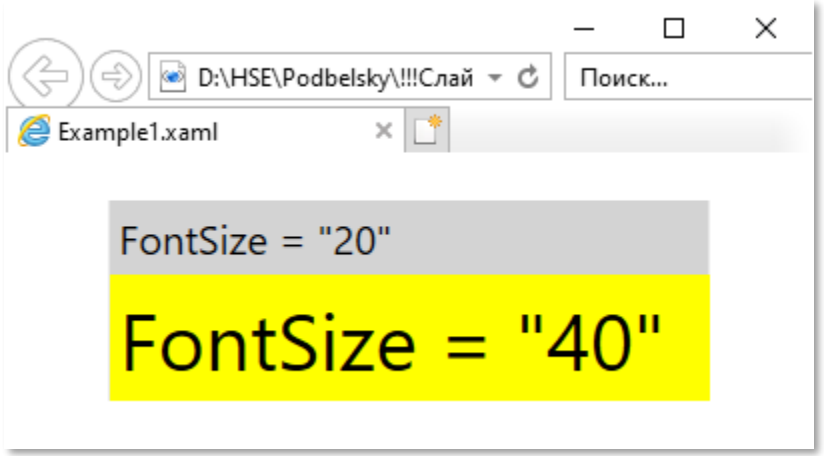
Анимированные значения
Локальное значение
Шаблонные свойства
Методы установки значений для стиля
Значение по умолчанию

Пример: зависимые свойства

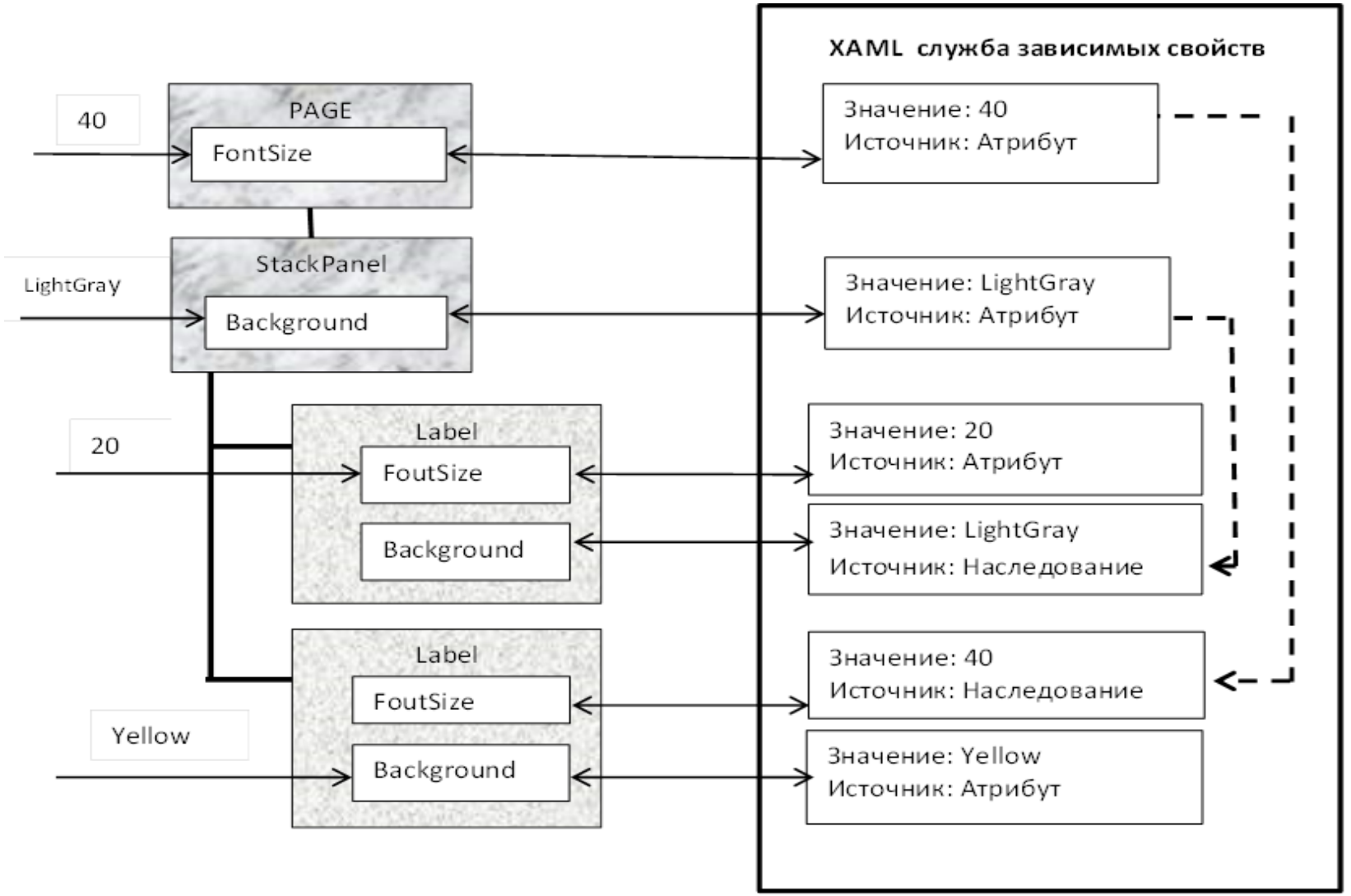
```
<?xml version="1.0" encoding="utf-8" ?>
<!-- Example1.xaml - зависимые свойства -->
<Page
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    FontSize="40" Width="300" Height="100">

    <StackPanel Background="LightGray">
        <Label FontSize="20">
            FontSize = "20"
        </Label>
        <Label Background="Yellow">
            FontSize = "40"
        </Label>
    </StackPanel>

</Page>
```

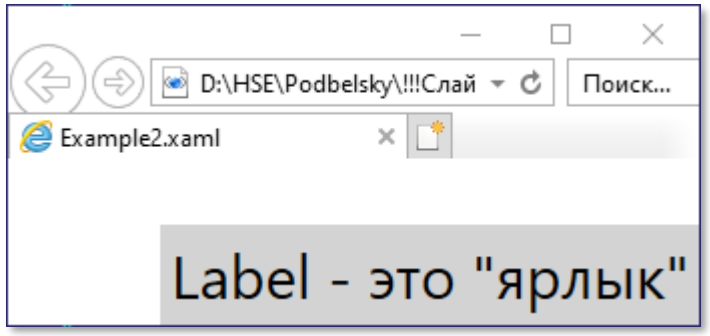


Служба поддержки зависимых свойств



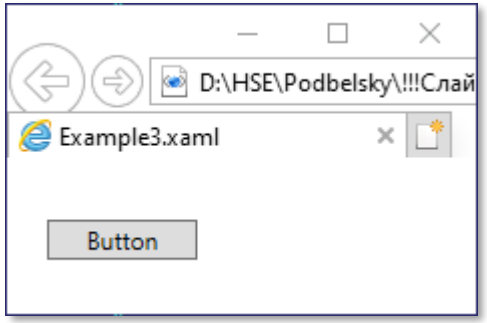
Присоединенные свойства (Attached Properties)

```
<!-- Example2.xaml  ПРИСОЕДИНЕННЫЕ СВОЙСТВА -->
<Canvas
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation">
    <Label FontSize="30" Background="LightGray"
Canvas.Right="0" Canvas.Bottom="0">
        Label - это "ярлык"
    </Label>
</Canvas>
```



Конвертеры типов для значений атрибутов

```
<!-- Example3.xaml -  Преобразование типов -->
<Grid
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation">
    <Button Content="Button" HorizontalAlignment="Left"
VerticalAlignment="Top" Width="75">
        <Button.Margin>
            <Thickness Left="20" Top="30" Right="20" Bottom="10"/>
        </Button.Margin>
    </Button>
</Grid>
```



Конвертеры типов для значений атрибутов

```
<Button Margin="20, 30, 20, 10" Content="Button"/>
<Button Margin="10" Content="Button"/>
```

```
<Label FontSize="30" Background="LightGray">
```

System.Windows.Media.Brush - тип свойства **Background**

Расширение разметки (Markup Extensions)

{*Имя_Расширения Аргументы*}

- {x:Bind} – компилируемая привязка данных (Windows 10);
- {Binding} – привязка данных;
- {StaticResource} – ссылка на ресурс в ResourceDictionary;
- {ThemeResource} – в ThemeResource - ссылки на ключи ресурсов в зависимости от активной темы;
- {TemplateBinding} – привязка в шаблонах элементов управления;
- {RelativeSource} – указание относительного источника внутри дерева XAML (Self / TemplatedParent);
- {CustomResource} – пользовательская ссылка на ресурс.

https://docs.microsoft.com/ru-ru/dotnet/desktop/xaml-services/markup-extensions-overview

Расширение разметки (Markup Extensions)

Расширения разметки, не принадлежащие пространству имен, выбираемому по умолчанию:

x:Type
x:Static
x:Array
x:Reference
x:Null

https://docs.microsoft.com/ru-ru/dotnet/desktop/xaml-services/namespace-language-features

Расширение разметки x:Static (Markup Extensions)

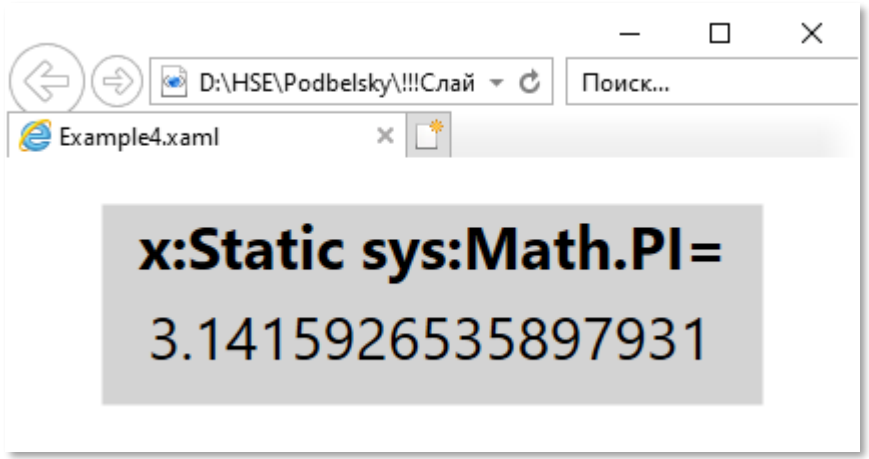
Расширение разметки {x:Static} в XAML атрибуте:

```
<object property="{x:Static
        prefix:typeName.staticMemberName}" .../>
```

staticMemberName – имя статического члена
typeName – имя типа
prefix – указание на пространство имен
{ и } - другие атрибуты (свойства элемента)

Расширение разметки x:Static (Markup Extensions)

```
<!-- Example4.xaml -  Расширение разметки -->
<Page Width="330" Height="100" Background="LightGray"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:sys="clr-namespace:System;assembly=mscorlib" >
    <StackPanel  HorizontalAlignment="Center" >
        <TextBlock Text="x:Static sys:Math.PI=" FontSize="30"
FontWeight="Bold"/>
        <Label Content="{x:Static sys:Math.PI}" FontSize="30"/>
    </StackPanel>
</Page>
```



Ошибка в расширении разметки

```
<TextBlock Text="{x:Static sys:Math.PI}" FontSize="30"/>
```

Результат - необработанное исключение:
System.Windows.Markup.XamlParseException -> System.ArgumentException: "3,14159265358979" не является допустимым значением для свойства "Text".

```
public string Text { get; set; }
```

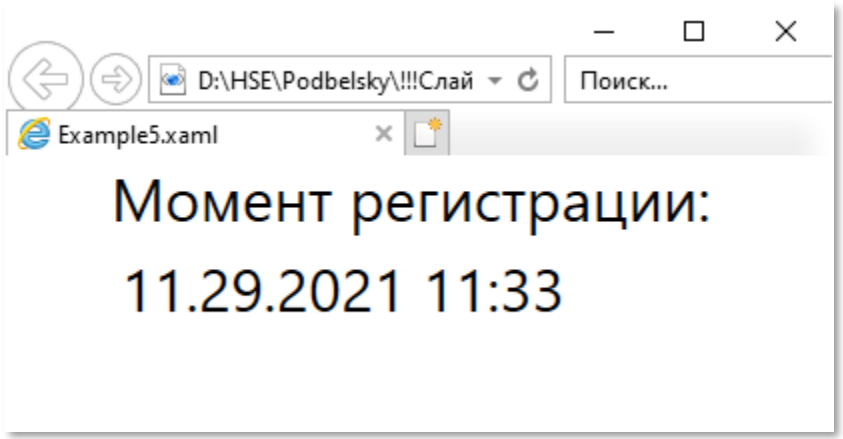
Корректная разметка:

```
public object Content { get; set; }
```

```
<Label Content="{x:Static sys:Math.PI}" FontSize="30"/>
```

Расширение разметки (Markup Extensions)

```
<!-- Example5.xaml - Статические Дата и время в XAML -->
<Page
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:sys="clr-namespace:System;assembly=mscorlib" >
    <StackPanel HorizontalAlignment="Center">
        <TextBlock Text="Момент регистрации: " FontSize="30" />
        <Label Content="{x:Static sys:DateTime.Now}"
FontSize="30"/>
    </StackPanel>
</Page>
```



Привязка данных (Data Binding)

Цель:

цель воздействия связи (приемник)

свойство объекта-приемника

Источник:

источник связи

свойство объекта-источника

Источник связи указывается в приемнике:

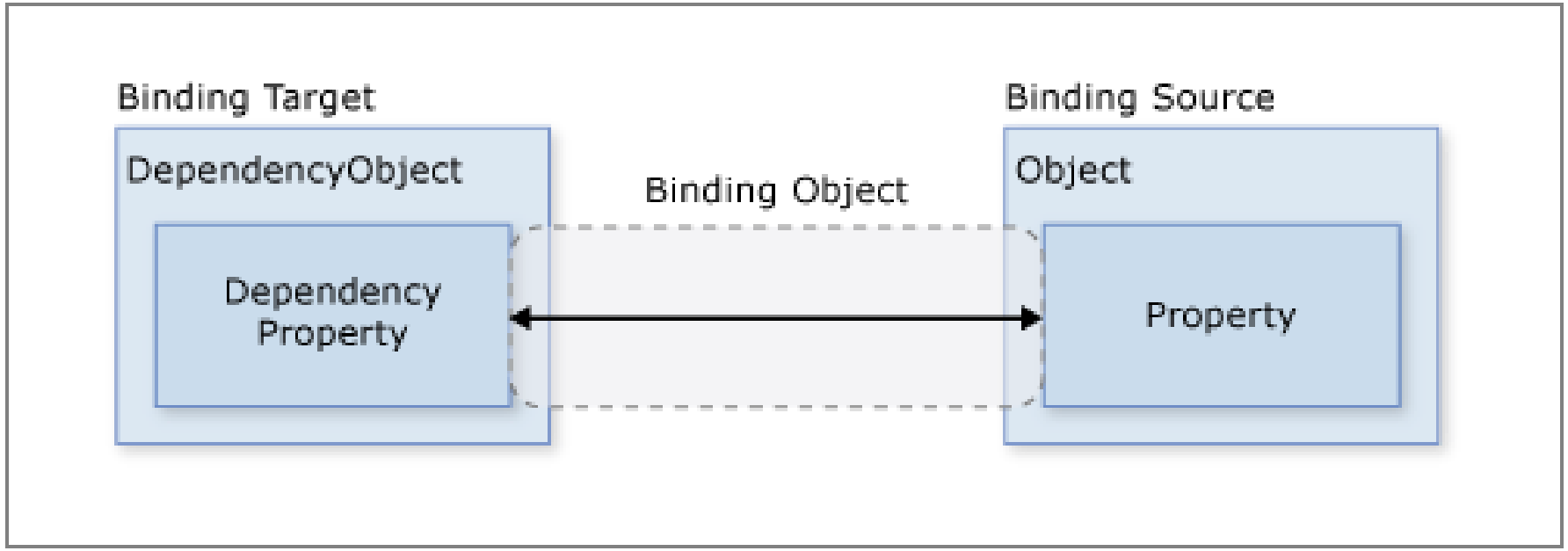
{ Binding

ElementName=*имя_XAML_источника*,

Path=*свойство* }

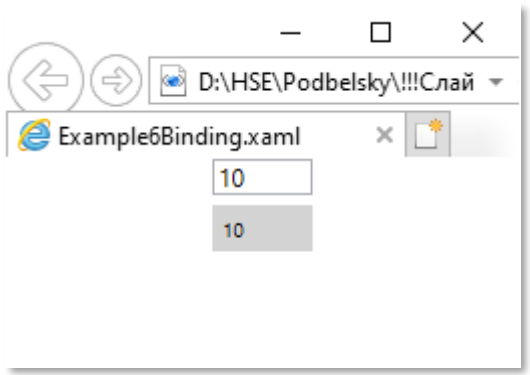
https://docs.microsoft.com/ru-ru/dotnet/desktop/wpf/data/

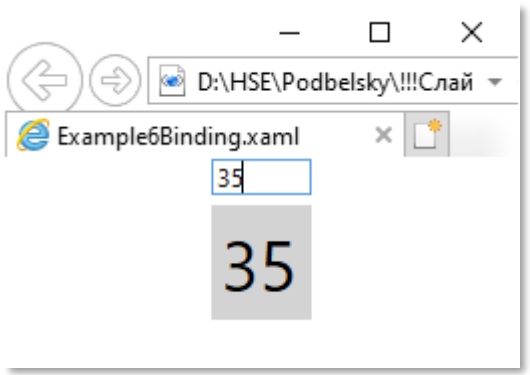
Привязка данных (Data Binding)

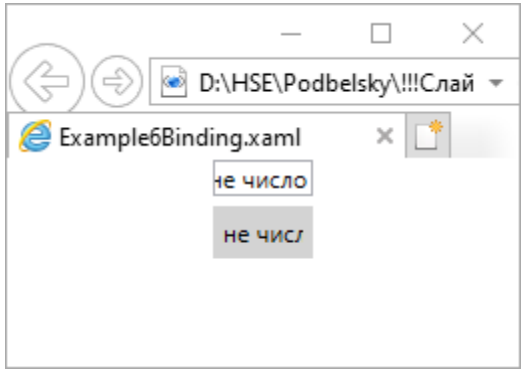


Привязка данных (Data Binding)

```
<!-- 6Binding.xaml - Data Binding -->
<StackPanel
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation">
    <TextBox Name="sourceInfo" Width="50">10</TextBox>
    <Label Margin="5" Width="50" Background="LightGray"
      FontSize="{Binding ElementName=sourceInfo, Path=Text}"
      Content="{Binding ElementName=sourceInfo, Path=Text}" />
</StackPanel>
```

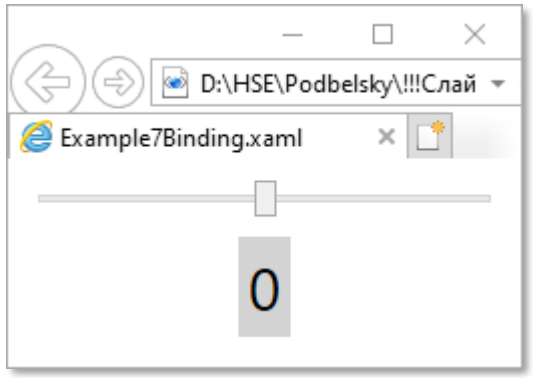


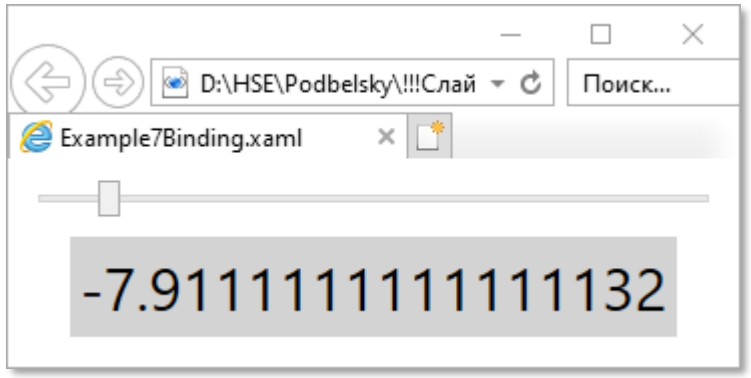




Привязка данных (Data Binding)

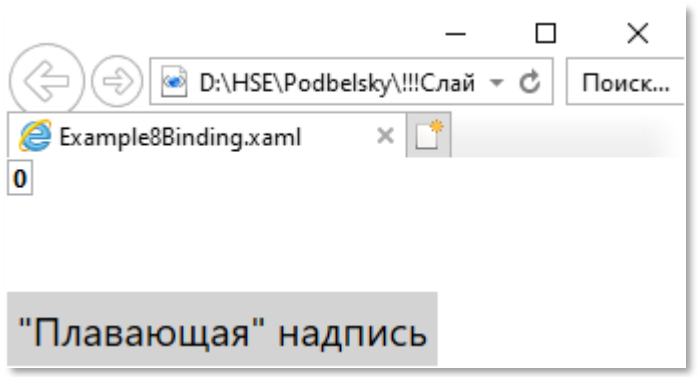
```
<!-- 7Binding.xaml - Data Binding -->
<StackPanel
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation">
    <Slider Name="slider" Minimum="-10" Maximum="10"
  Orientation="Horizontal" Margin="10" />
    <Label FontSize="30" Background="LightGray"
        HorizontalAlignment="Center">
        <Label.Content>
            <Binding ElementName="slider" Path="Value" />
        </Label.Content>
    </Label>
</StackPanel>
```

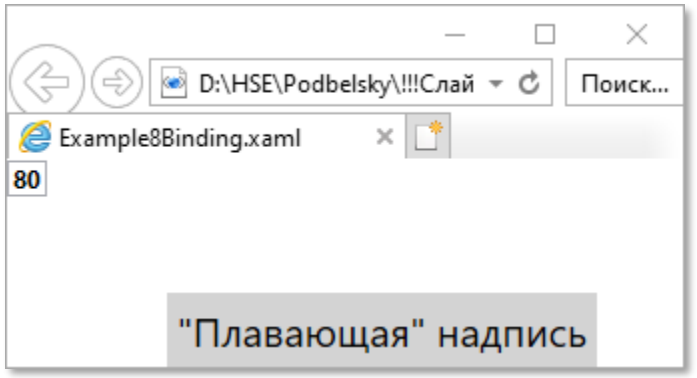




Привязка данных (Data Binding)

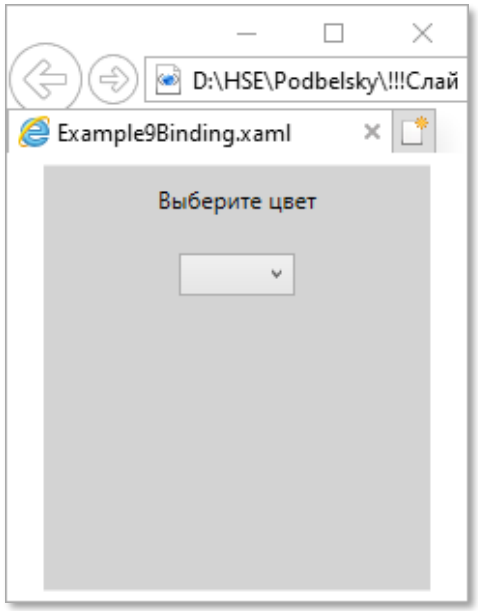
```
<!-- 8Binding.xaml - Плавающая надпись -->
<Canvas
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation">
    <TextBox Name="input" FontWeight="Bold">0</TextBox>
    <Label FontSize = "20" Background="LightGray"
      Canvas.Left="{Binding ElementName=input, Path=Text}"
      Canvas.Bottom="0">
        "Плавающая" надпись
    </Label>
</Canvas>
```

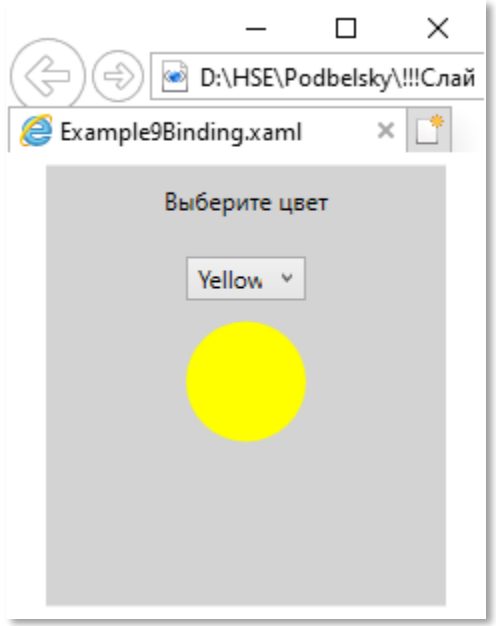




Привязка данных (Data Binding)

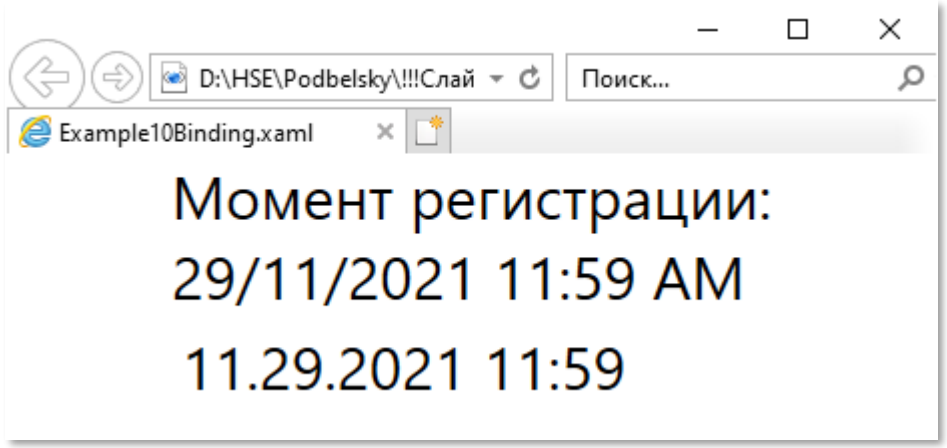
```
<!-- 9Binding.xaml - Светофор -->
<Page Width="200" Height="220" Background="LightGray"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation">
    <StackPanel>
        <TextBlock Text="Выберите цвет" Margin="10"
HorizontalAlignment="Center"/>
        <ComboBox Name="comboBox" Margin="50,10" Width="60">
            <ComboBoxItem Content="Green"/>
            <ComboBoxItem Content="Yellow" />
            <ComboBoxItem Content="Red" />
        </ComboBox>
        <Ellipse Width="60" Height="60"
    Fill="{Binding ElementName=comboBox,
        Path=SelectedItem.Content}" />
    </StackPanel>
</Page>
```



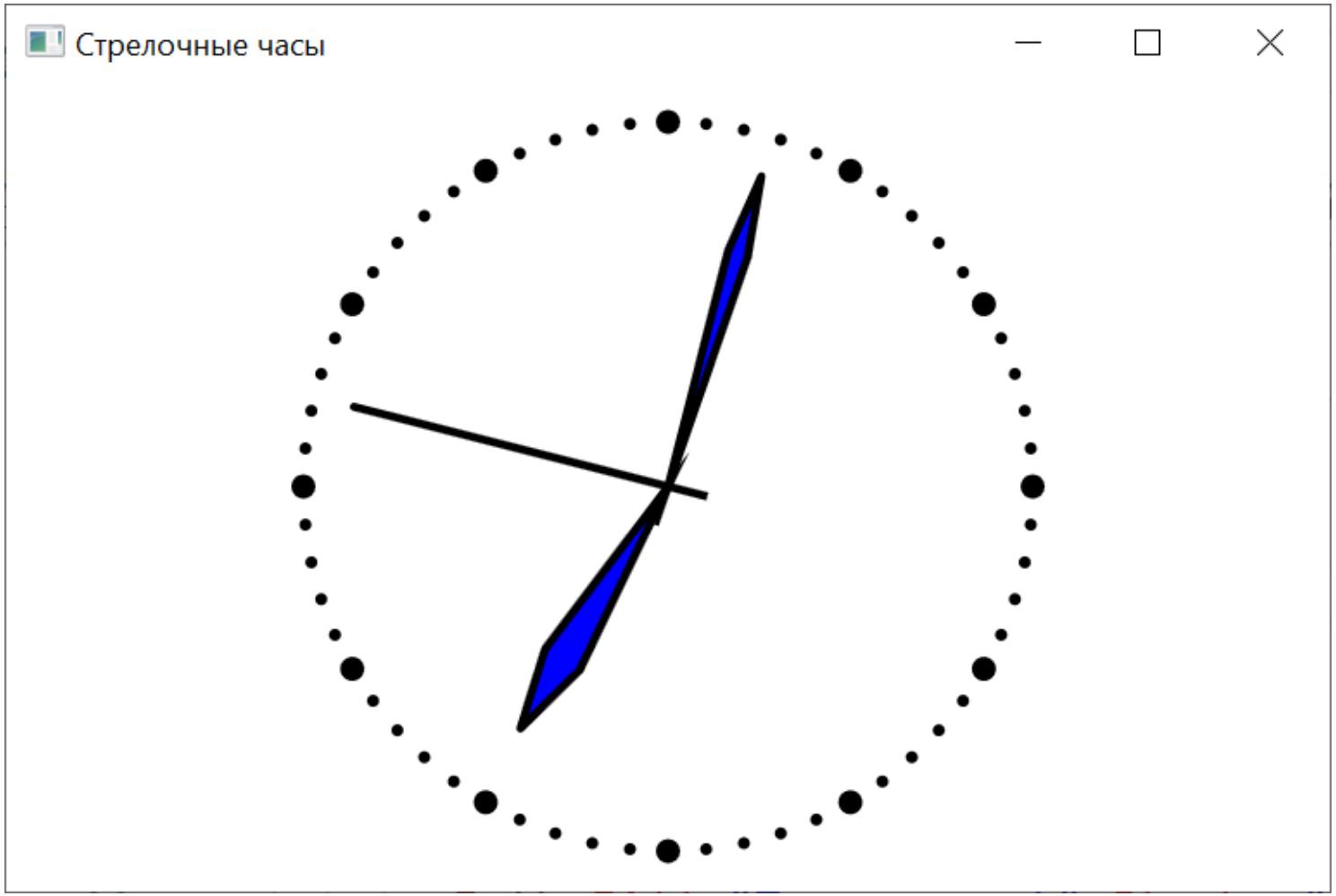


Форматирование даты и времени

```
<!-- 10Binding.xaml - DateTime format -->
<Page
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:sys="clr-namespace:System;assembly=mscorlib" >
    <StackPanel HorizontalAlignment="Center">
        <TextBlock Text="Момент регистрации:" FontSize="30" />
        <TextBlock Text="{Binding Source={x:Static
sys:DateTime.Now}, StringFormat='dd/MM/yyyy hh:mm tt'}"
FontSize="30"/>
        <Label Content="{x:Static sys:DateTime.Now}" FontSize="30"/>
    </StackPanel>
</Page>
```

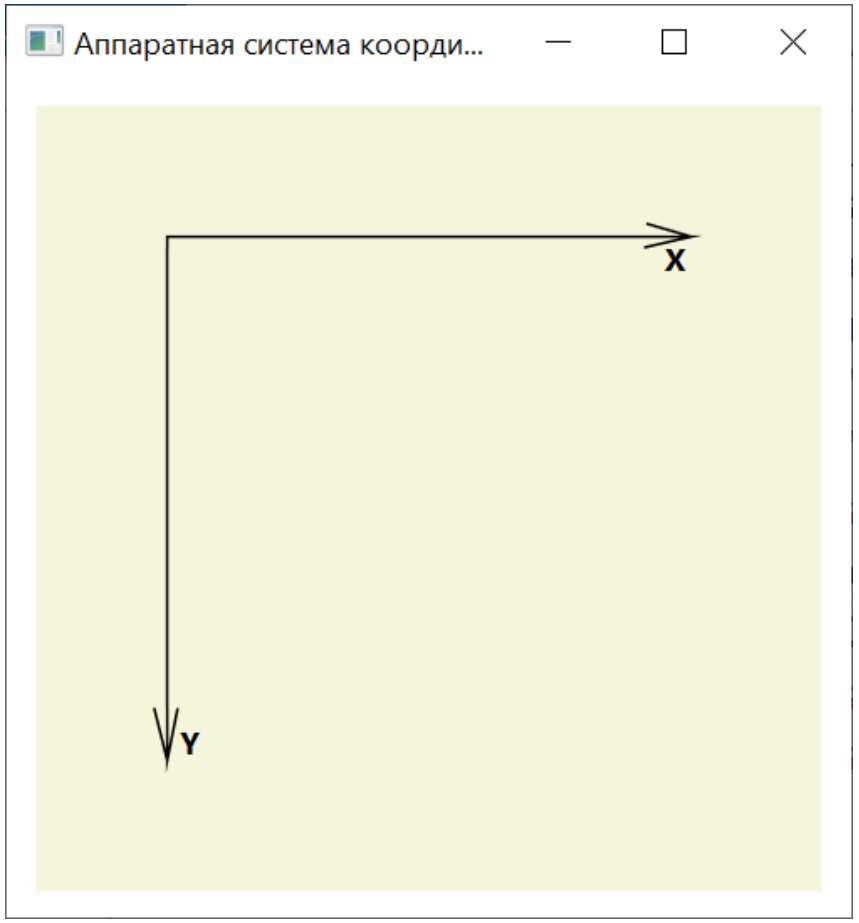


Проект «Стрелочные часы»



Системы координат в компьютерной графике

- Аппаратная система координат (device coordinate system) или система координат устройства, или физическая система координат (device-independent pixel)



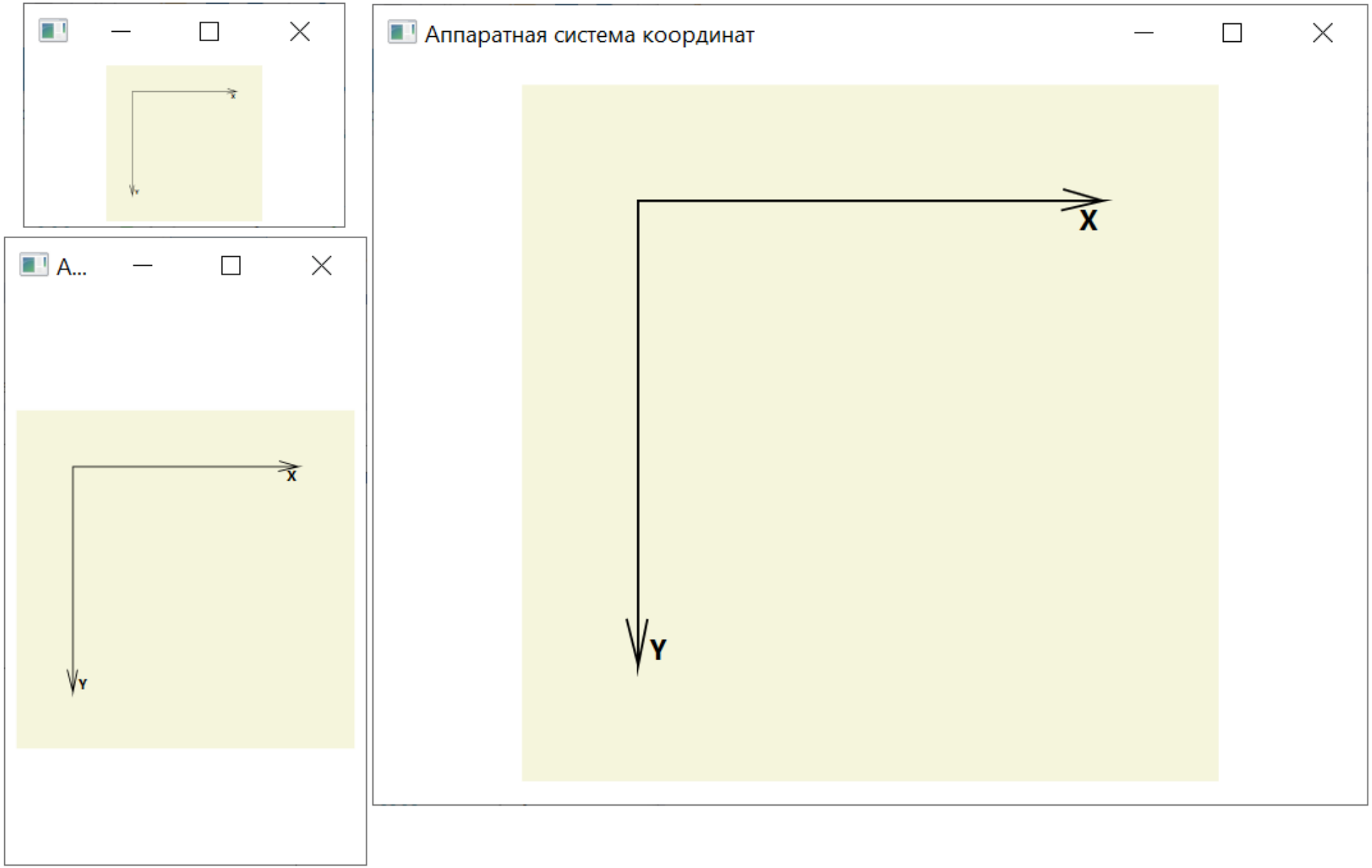
Аппаратные координаты (XAML разметка)

```
<Window x:Class="Аппаратные_координаты.MainWindow"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="Аппаратная система координат" FontWeight="Bold"
      Height="370" Width="350">
  <Viewbox Stretch="Uniform">
    <Canvas x:Name="Pole" Margin="10" Height="300" Width="300" Background ="Beige">
      <Polyline Stroke="Black" StrokeThickness="1"
        Points="45,230 50,250 53,235 50,250 50,50 250,50 237,53 250,50 233,45" />
      <TextBlock Text="X" Canvas.Left ="240" Canvas.Top="50"/>
      <TextBlock Text="Y" Canvas.Left ="55" Canvas.Top="235"/>
    </Canvas>
  </Viewbox>
</Window>
```

Элементы перечисления Stretch определяющие значения свойства Stretch

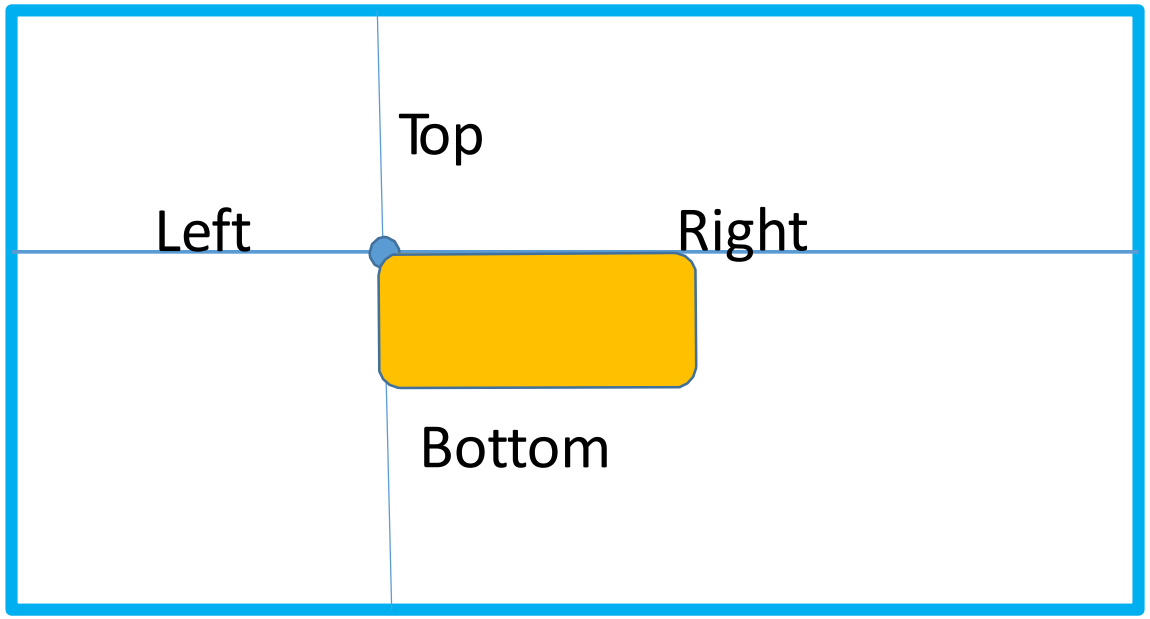
Элемент перечисления	Описание
None	Содержимое визуализируется в заданном масштабе и не изменяет размеры при изменении размеров контейнера
Fill	Содержимое масштабируется без сохранения пропорций, занимая полностью выделяемое контейнером пространство
Uniform (установлено по умолчанию)	Содержимое выводится целиком и масштабируется, сохраняя исходные пропорции. Слева и справа (или сверху и снизу) от изображения могут остаться незаполненные фрагменты контейнера.
UniformToFill	Содержимое масштабируется с сохранением пропорций до полного заполнения контейнера (часть изображения, выходящая за пределы контейнера отсекается)

Особенности элемента Viewbox с атрибутом Stretch="Uniform"



Присоединенные свойства холста

- **Canvas.Left** – смещение элемента от левой границы холста,
- **Canvas.Right** – смещение элемента от правой границы холста,
- **Canvas.Top** – смещение элемента от верхней границы холста,
- **Canvas.Bottom**– смещение элемента от нижней границы холста,
- **Canvas.ZIndex** – «уровень» элемента в стопке перекрывающихся элементов.



Классы, производные от Shape

- **Ellipse –** эллипс;
- **Line –** отрезок прямой;
- **Path –** траектория;
- **Polygon –** многоугольник;
- **Polyline –** ломаная линия;
- **Rectangle –** прямоугольник.

Класс **Shape** является потомком классов **UIElement** и **FrameworkElement.**

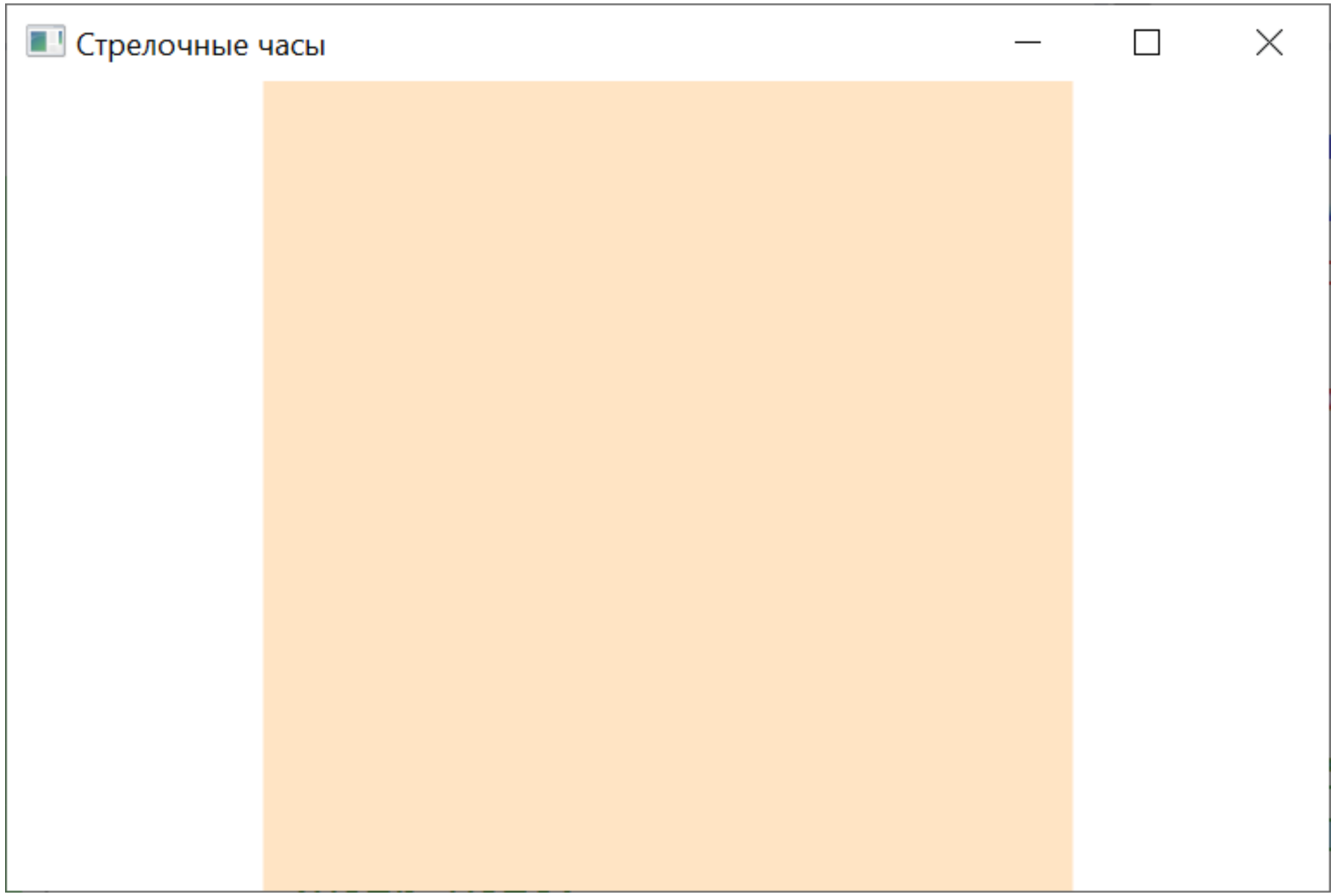
В фоновом коде требуется директива:

```
using System.Windows.Shapes;
```

Начальная разметка проекта

```
<Window x:Class="Стрелочные_часы.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Стрелочные часы" Height="350" Width="525">
  <Viewbox>
    <Canvas Width="200" Height="200" Background="Bisque">
. . . . . .
    </Canvas>
  </Viewbox>
</Window>
```

Отображение пустого холста в главном окне



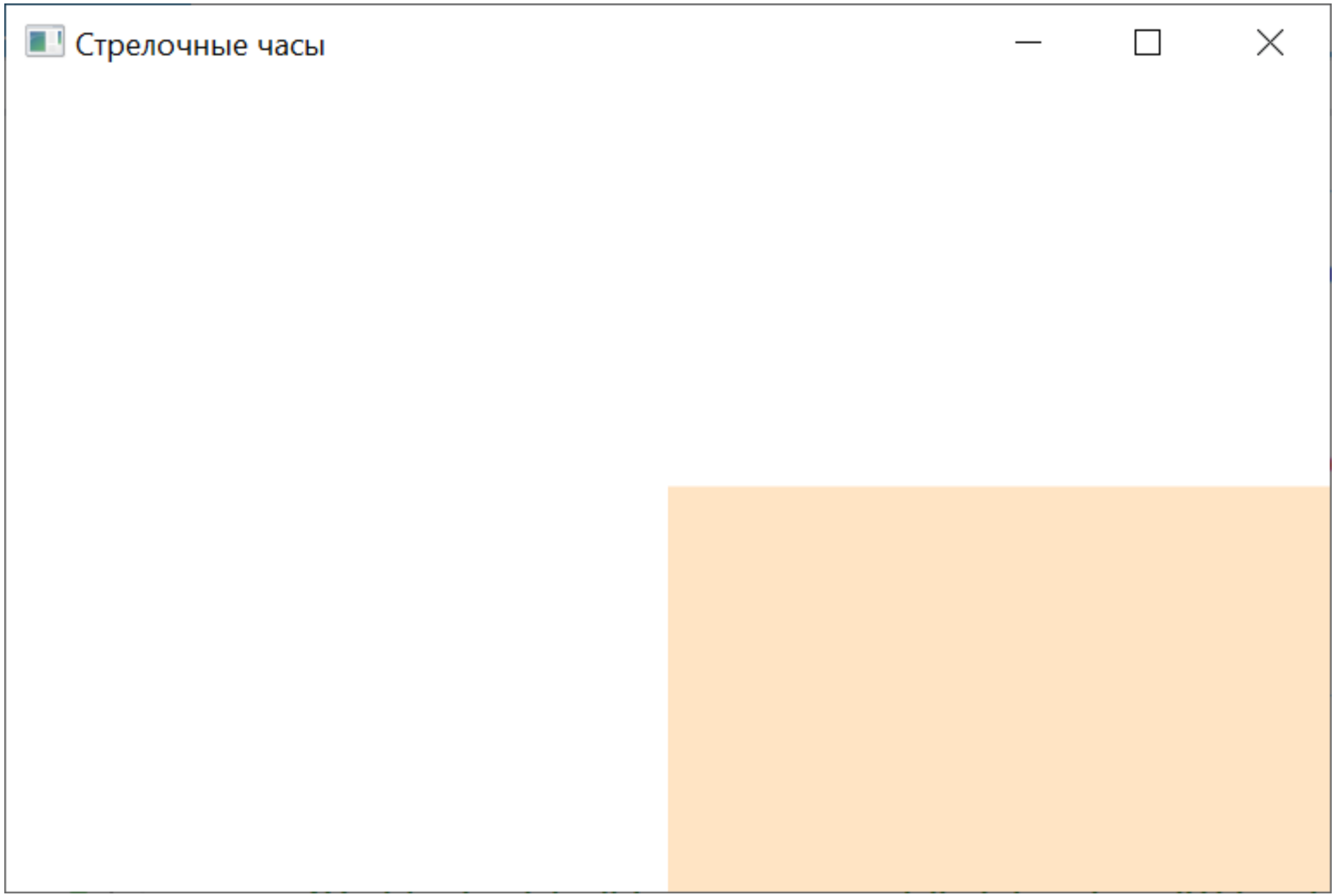
Свойство RenderTransform и аффинные преобразования на плоскости

Класс	Описание
RotateTransform	Поворот
ScaleTransform	Масштабирование
SkewTransform	Скашивание (наклон, скос)
TranslateTransform	Сдвиг (перемещение)
MatrixTransform	Матричное преобразование
CompositeTransform	Составное (сложное) преобразование
TransformGroup	Группа преобразований

Элемент-свойство Canvas.RenderTransform и элемент TranslateTransform

```
<Canvas Width="200" Height="200" Background="Bisque">
    <Canvas.RenderTransform>
      <TranslateTransform X="100" Y="100"/>
    </Canvas.RenderTransform>
  </Canvas>
```

Результат отображения



Класс Path, свойство Path.Data и классы, производные от класса Geometry

Класс	Описание
LineGeometry	Отрезок прямой линии
EllipseGeometry	Эллипс
RectangleGeometry	Прямоугольник
GeometryGroup	Объединение нескольких геометрических фигур
CombinedGeometry	Комбинирование двух геометрических фигур
PathGeometry	Траектория – совокупность геометрических фигур
StreamGeometry	Немодифицируемый вариант PathGeometry

Формат декларации XAML-элемента Path

<Path *атрибуты*>

<Path.Data>

<*Класс_геометрии* атрибуты>

</*Класс_геометрии*>

</Path.Data>

</Path>

Атрибуты элемента Path:

Stroke - цвет границы;

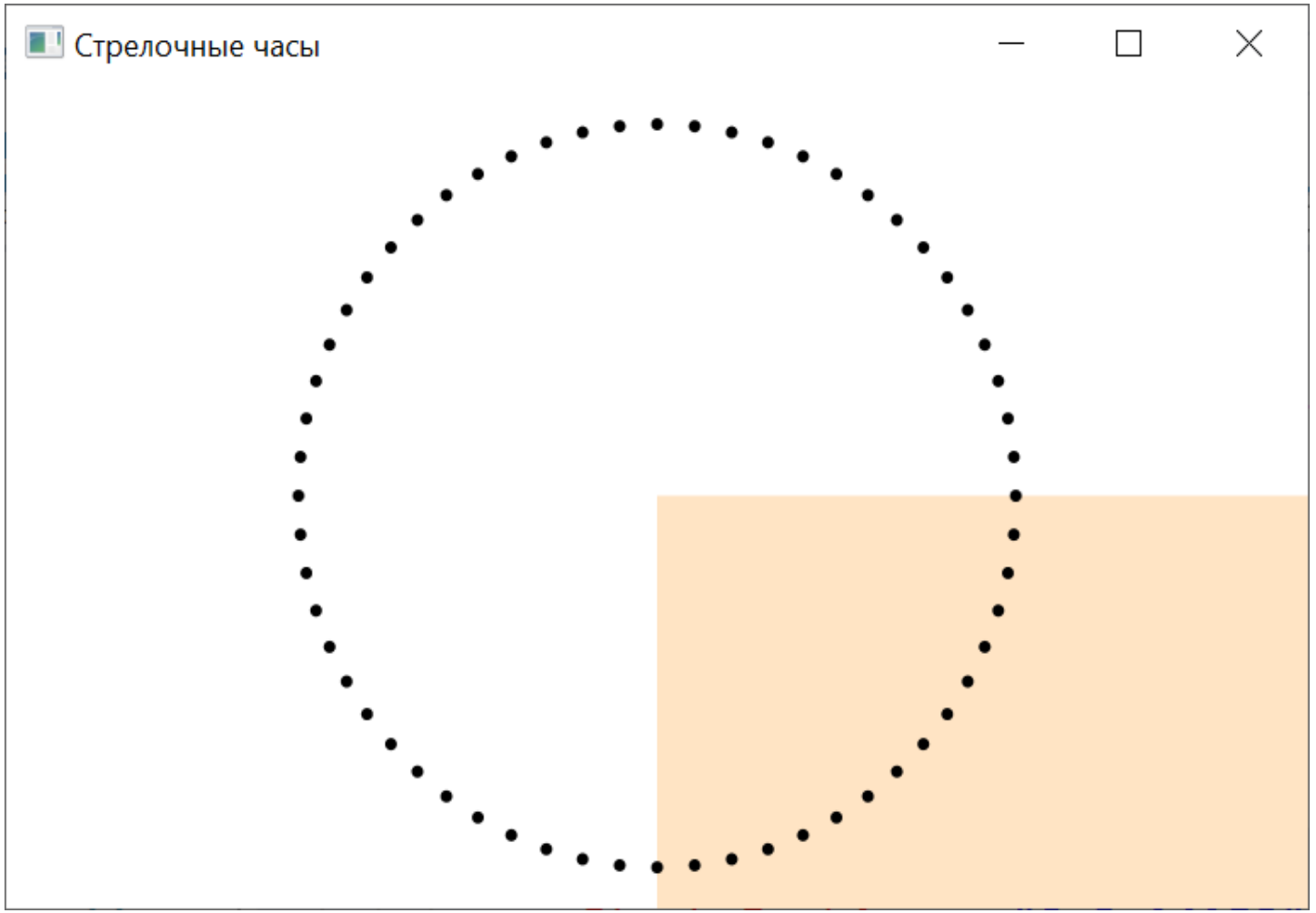
StrokeThickness - ширина (толщина) линии границы;

Fill - цвет внутренней окраски фигуры.

Свойства элемента EllipseGeometry

Свойство	Тип	Описание
RadiusX	Double	Радиус вдоль оси абсцисс (полуось эллипса)
RadiusY	Double	Радиус вдоль оси ординат (полуось эллипса)
Center	Point	Положение центра эллипса
Transform	Transform	Преобразование системы координат изображения

Минутные деления на циферблате



XAML разметка для минутных делений

```
<Path Fill="Transparent" Stroke="Black" StrokeThickness="3"
                StrokeDashArray="0    3.14159" StrokeDashCap ="Round">
    <Path.Data>
      <EllipseGeometry RadiusX="90" RadiusY="90"/>
    </Path.Data>
 </Path>
```

Абсолютная длина «минутной» дуги:

$2*\pi*R/60 = 2*\pi*90/60 = 3*\pi$.

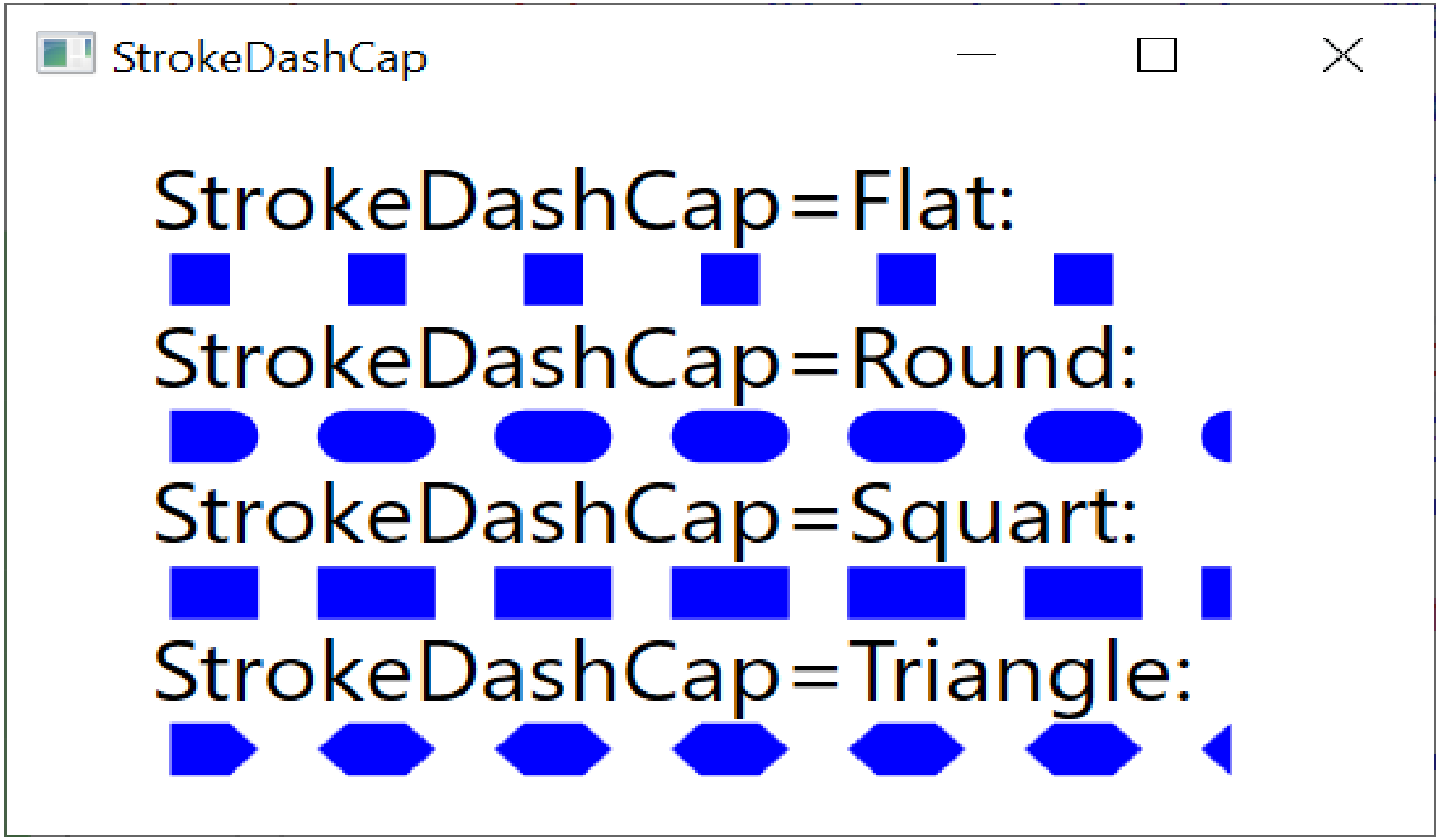
Относительное значение: $3*\pi/StrokeThickness = \pi = 3.14159$.

https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.shapes.shape.strokedasharray

Перечисление PenLineCap

Элемент	Описание
Flat	Окончание и начало изображения линии совпадает с ее последней точкой (этот элемент перечисления выбирается по умолчанию)
Round	К началу и к концу линии добавляется закругление (полукруг) с диаметром, равным ширине линии
Square	К началу и к концу линии добавляется прямоугольник, одна сторона которого равна ширине линии, вторая – в два раза меньше
Triangle	К началу и к концу линии добавляется равнобедренный прямоугольный треугольник, основание которого равно ширине линии

Штриховые линии (StrokeDashArray = "1, 2")



Часовые деления по окружности

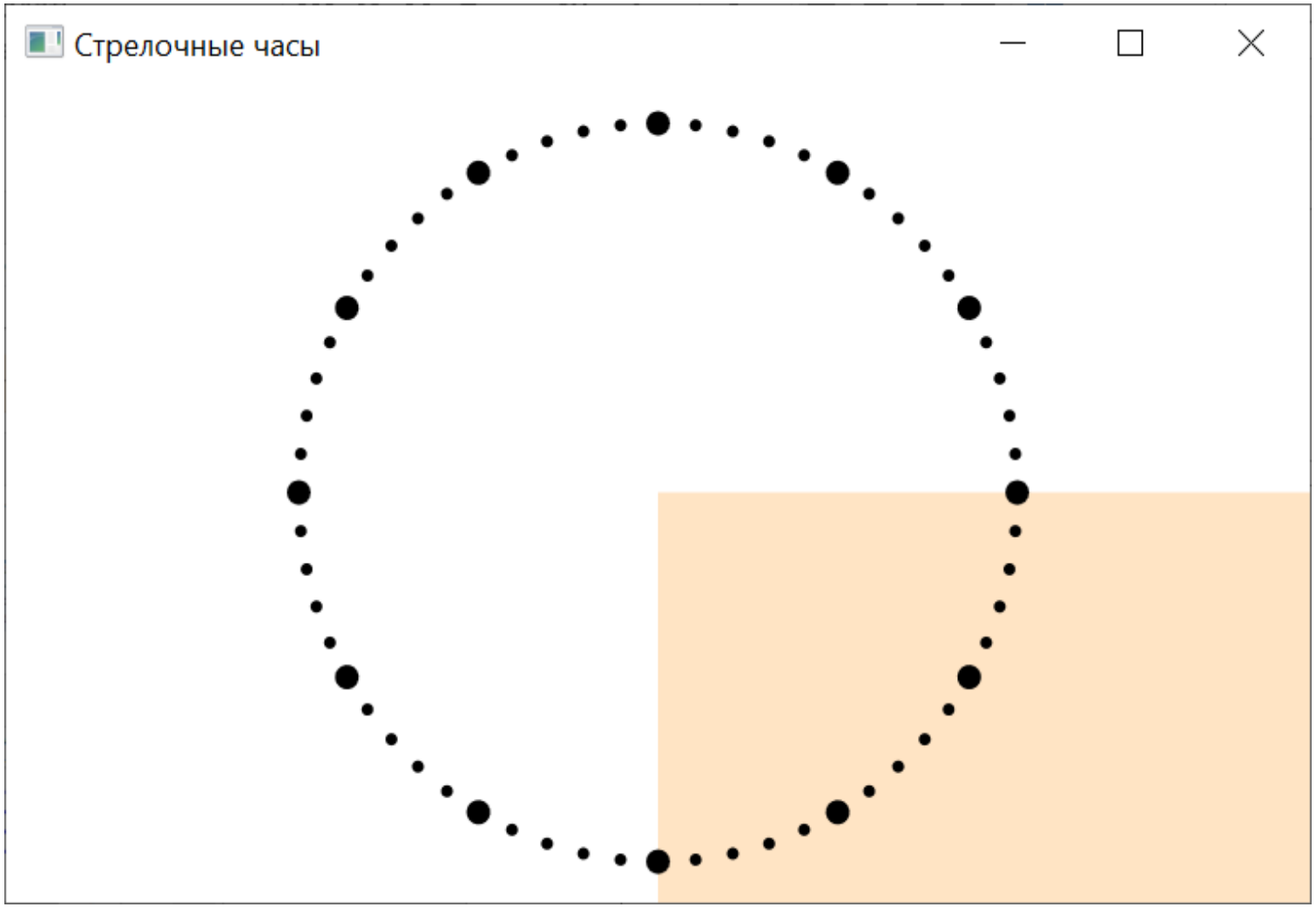
```
<Path Fill="Transparent" Stroke="Black" StrokeThickness="6"
                StrokeDashArray="0 7.85398" StrokeDashCap ="Round">
        <Path.Data>
                <EllipseGeometry RadiusX="90" RadiusY="90"/>
        </Path.Data>
</Path>
```

Абсолютная длина «часовой» дуги:

$2*\pi*R/12 = 2*\pi*90/12 = 15*\pi$.

Относительное значение: $15*\pi/\text{StrokeThickness} = 15*\pi/6 = 7.85398$.

Циферблат часов без стрелок



Декларация секундной стрелки

```
<!-- Секундная стрелка на 12 часов -->
<Path Data="M 0 10 L 0 -80"    StrokeEndLineCap="Round"
                Stroke="Black" StrokeThickness="2" >
    <Path.RenderTransform>
            <RotateTransform x:Name="rotateSecond"/>
    </Path.RenderTransform>
</Path>
```

Команды мини-языка разметки траекторий

Команда	Описание
M или m	Формат команды: “M x,y” (Move) – определяет точку начала нового фрагмента траектории; x, y – значения координат точки начала.
L или l	Формат команды: “L x,y” (Line) – провести отрезок прямой из «текущей» точки в точку, заданную параметрами-координатами x,y
Z или z	Формат команды: “Z” Замыкание фрагмента траектории - завершает декларацию фрагмента, соединяя его конец с началом.

Геометрия изображений стрелок часов

<!-- Секундная стрелка на 12 часов: -->

Data="M 0 10 L 0 -80"

<!-- Минутная стрелка на 12 часов: -->

Data="M 0 -80 L 2.5 -60 0 0 -2.55 -60 0 -80"

<!-- Часовая стрелка на 12 часов: -->

Data="M 0 -70 L 5 -50 L 0 0 L -5 -50 L 0 -70“

Команды:

M 0 -80

L 2.5 -60

L 0 0

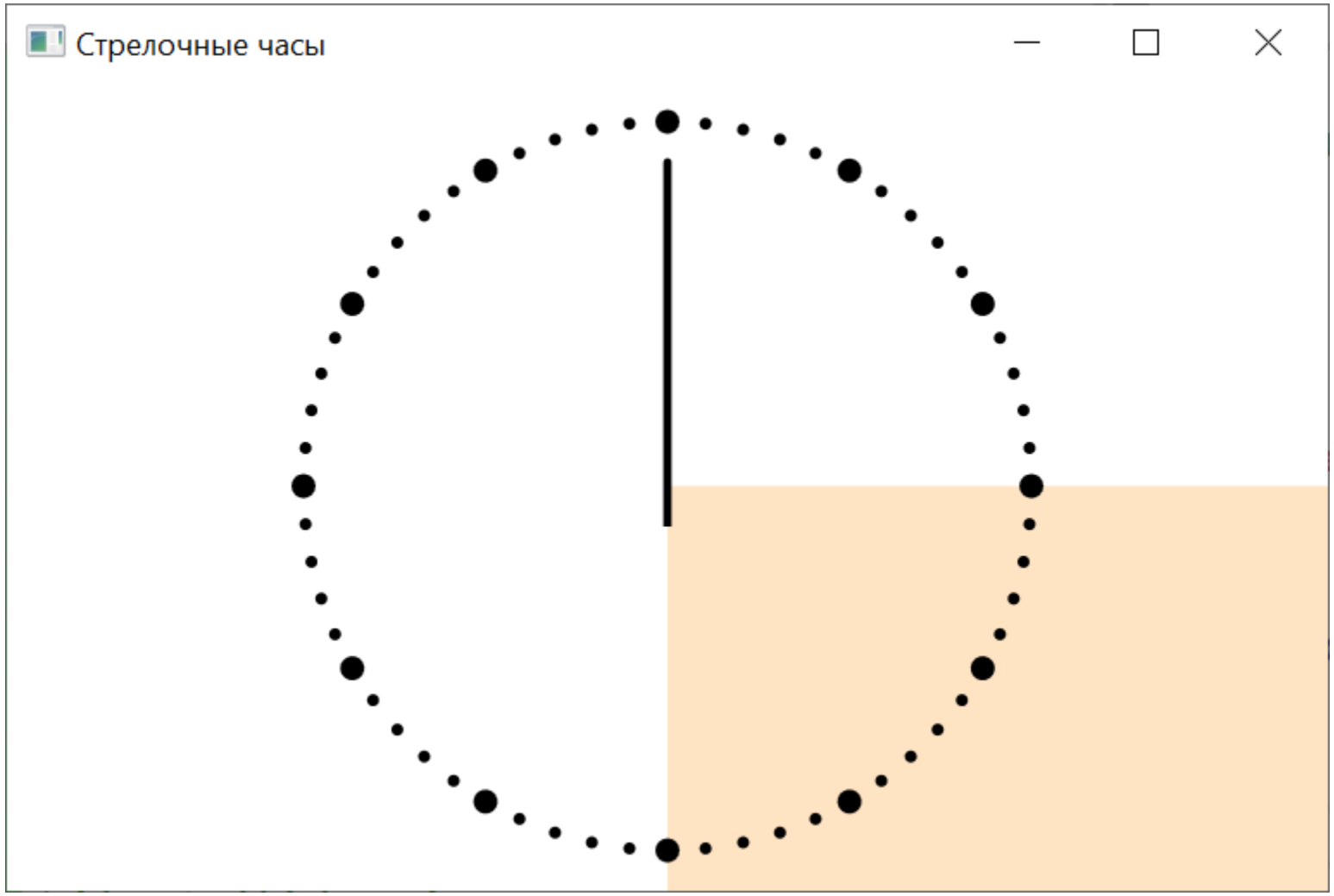
L -2.55 -60

L 0 -80

Примечание: Разделять координаты точек можно запятыми.

- Синтаксис разметки пути.
- https://msdn.microsoft.com/ru-ru/library/ms752293(v=vs.110).aspx
- Path Markup Syntax.
- https://msdn.microsoft.com/ru-ru/library/cc189041(v=vs.95).aspx

Циферблат часов с секундной стрелкой



Код для движения секундной стрелки

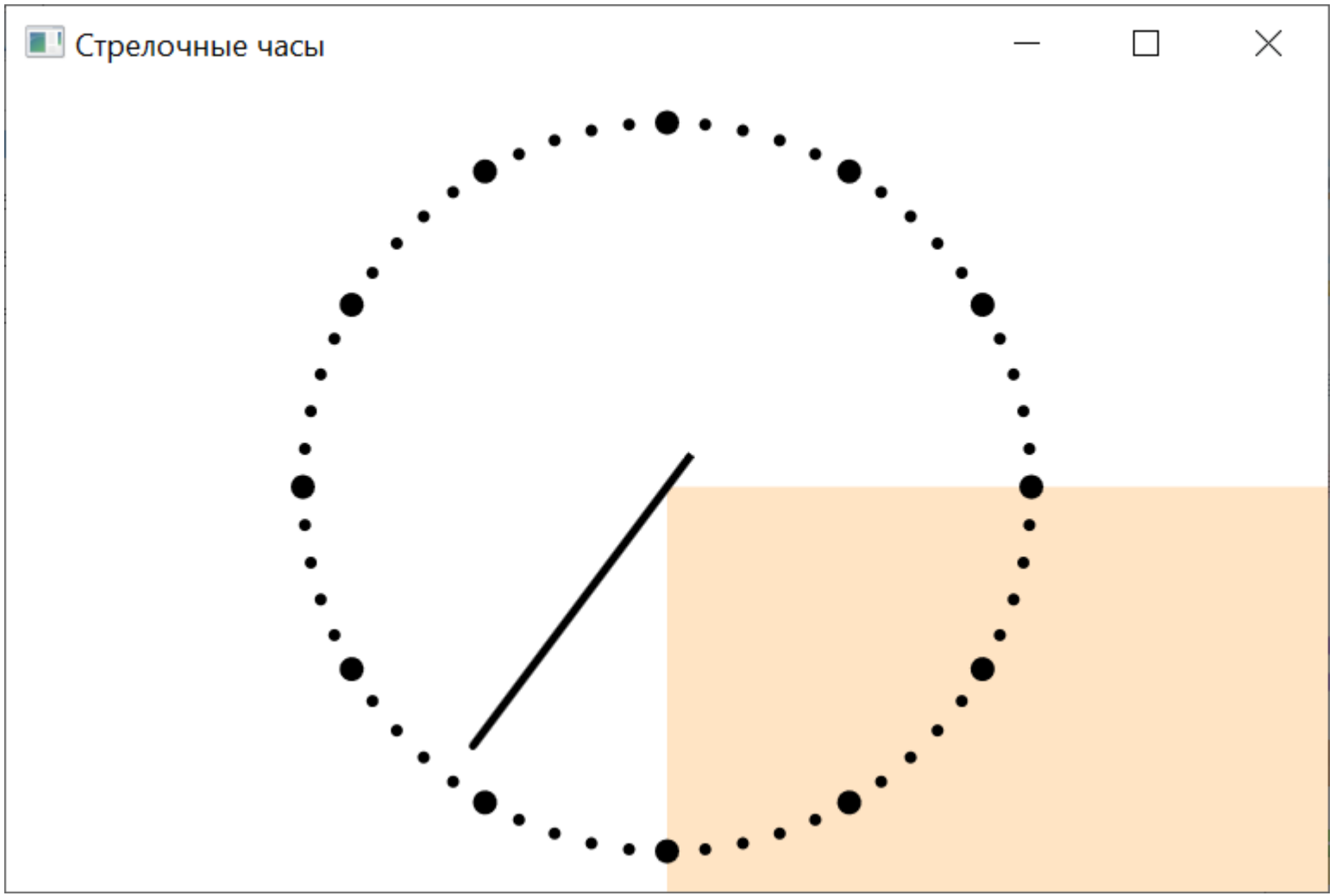
```
public partial class MainWindow : Window {
  public MainWindow() { InitializeComponent();
  CompositionTarget.Rendering += (ss, ee) => {
     DateTime dt = DateTime.Now;
     rotateSecond.Angle = 6 * (dt.Second + dt.Millisecond / 1000.0);
   };

  }
}
```

Вариант для «прыгающей» стрелки:
rotateSecond.Angle = 6 * dt.Second;

Движение секундной стрелки

Проект «Часы_0»



Минутная стрелка на 12 часов

```
<Path StrokeEndLineCap="Round" Fill="Blue"
      Stroke="Black" StrokeThickness="2" Data="M
      0 -80 L 2.5 -60 0 0 -2.55 -60          0 -80">
    <Path.RenderTransform>
      <RotateTransform x:Name="rotateMinute"/>
    </Path.RenderTransform>
  </Path>
```

Команды:

M 0 -80

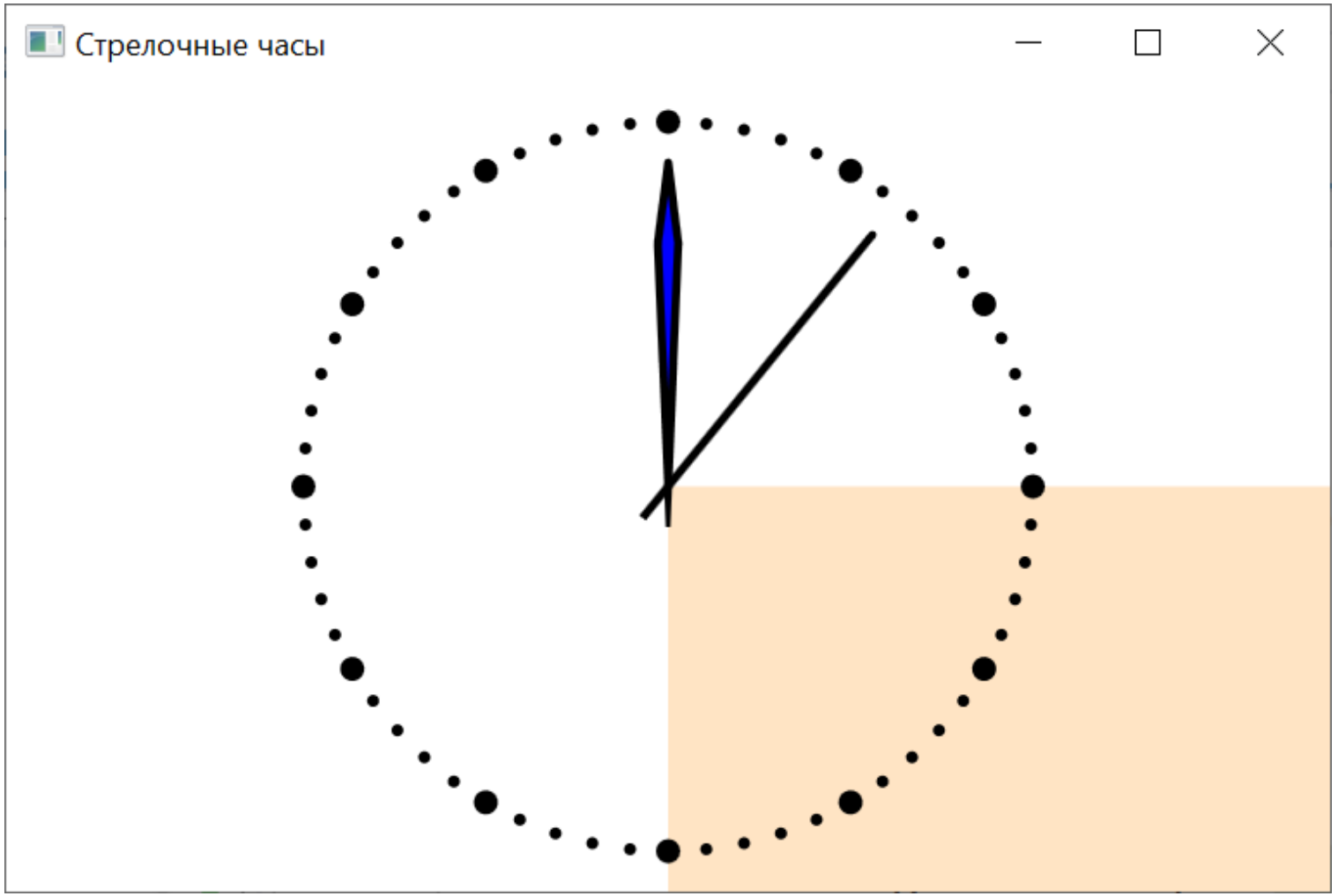
L 2.5 -60

L 0 0

L -2.55 -60

L 0 -80

Минутная стрелка на 12 часов



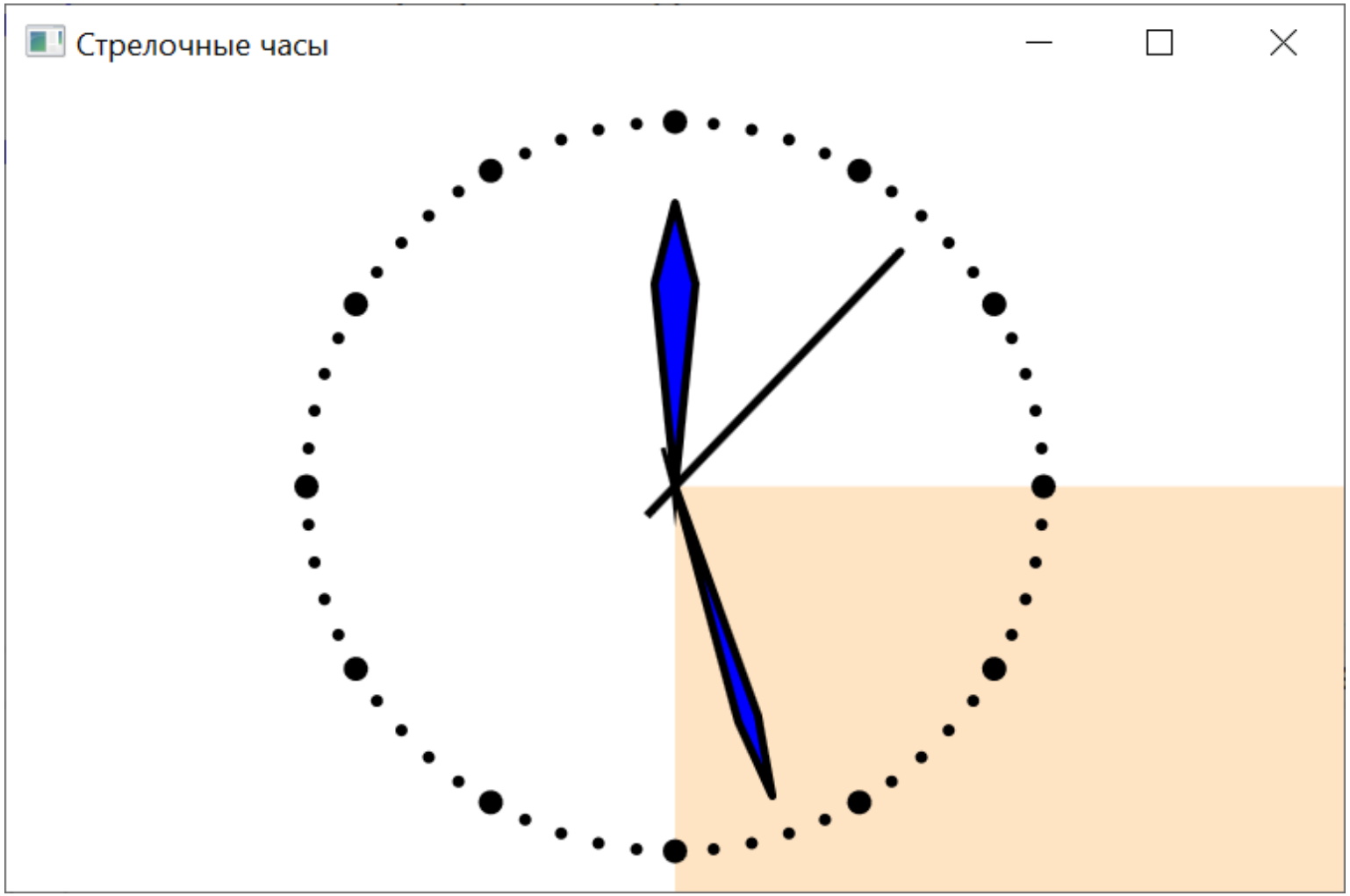
Текст обработчика событий с учетом минутной стрелки

```
CompositionTarget.Rendering += (ss, ee) => {
    DateTime dt = DateTime.Now;
    rotateSecond.Angle = 6 * (dt.Second + dt.Millisecond / 1000.0);
    rotateMinute.Angle = 6 * dt.Minute + rotateSecond.Angle / 60;
    };
```

Часовая стрелка на 12 часов

```
<Path StrokeEndLineCap="Round" Fill="Blue"
                Stroke="Black" StrokeThickness="2" Data="M
                0 -70 L 5 -50 L 0 0 L -5 -50 L 0 -70">
    <Path.RenderTransform>
                <RotateTransform x:Name="rotateHour"/>
    </Path.RenderTransform>
</Path>
```

Часовая стрелка на 12 часов



Полный текст обработчика событий

```
CompositionTarget.Rendering += (ss, ee) => {
    DateTime dt = DateTime.Now;
    rotateSecond.Angle = 6 * (dt.Second + dt.Millisecond / 1000.0);
    rotateMinute.Angle = 6 * dt.Minute + rotateSecond.Angle / 60;
    rotateHour.Angle = 30 * (dt.Hour % 12) + rotateMinute.Angle / 12;
};
```

Итоговый вариант часов