

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ



Федеральное государственное бюджетное образовательное  
учреждение высшего профессионального образования  
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Методические указания к лабораторной работе №9

## «Использование кистей в WPF-приложениях»

по дисциплине «Технологии разработки пользовательских интерфейсов»

Вичугов В.Н., доцент каф. АиКС

Томск 2012

# Элементы управления содержимым

## Элемент управления *ScrollView*

Данный элемент управления используется в случае, когда необходимо в ограниченной области окна вывести большой объем содержимого. Может содержать только один дочерний элемент (обычно – диспетчер компоновки).

Основные свойства:

**Width** – ширина;

**Height** – высота;

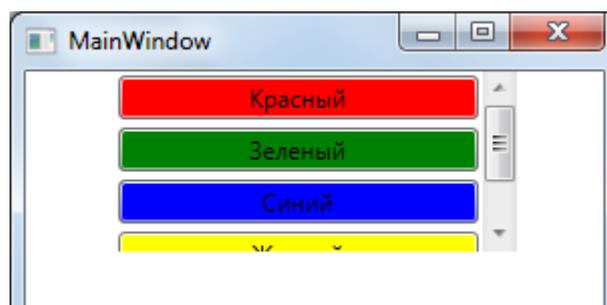
**HorizontalScrollBarVisibility** – отображение вертикальных полос прокрутки: **Visible** – отображаются, **Auto** – отображаются при необходимости, **Disabled** – не отображаются и прокрутка в данном направлении невозможна, **Hidden** – не отображаются, но прокрутка возможна (например, с помощью клавиш управления курсором);

**VerticalScrollBarVisibility** – отображение вертикальных полос прокрутки;

### Пример 1 Код XAML

```
<ScrollView Width="200" Height="90">
  <StackPanel Orientation="Vertical">
    <Button Background="Red">Красный</Button>
    <Button Background="Green">Зеленый</Button>
    <Button Background="Blue">Синий</Button>
    <Button Background="Yellow">Желтый</Button>
    <Button Background="Brown">Коричневый</Button>
  </StackPanel>
</ScrollView>
```

### Результат



## Элемент управления *Border*

Данный элемент управления используется для вывода закругленной границы вокруг других элементов управления. Может содержать только один дочерний элемент (обычно – диспетчер компоновки).

Основные свойства:

**BorderBrush** – кисть для рисования границы (в простейшем случае – цвет сплошной линии);

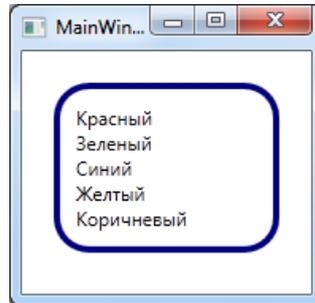
**BorderThickness** – толщина границы;

**CornerRadius** – радиус закругления границы; если указано одно число, то оно используется для всех углов границы, либо можно задать значение в формате «А,В,С,Д», где А – радиус закругления верхнего левого угла; В – верхнего правого, С – нижнего правого, D – нижнего левого.

### Пример 2 Код XAML

```
<Border BorderBrush="Navy" BorderThickness="4" CornerRadius="20" Padding="10" Margin="20">
  <StackPanel Orientation="Vertical" Background="White">
    <TextBlock>Красный</TextBlock>
    <TextBlock>Зеленый</TextBlock>
    <TextBlock>Синий</TextBlock>
    <TextBlock>Желтый</TextBlock>
    <TextBlock>Коричневый</TextBlock>
  </StackPanel>
</Border>
```

## Результат



## Элемент управления *GroupBox*

Данный элемент управления используется для группировки взаимосвязанных элементов управления. Может содержать только один дочерний элемент (обычно – диспетчер компоновки).

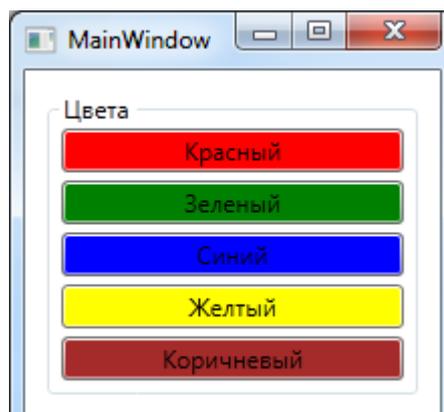
Основные свойства:

**Header** – заголовок элемента управления. Может быть текстовой строкой, либо любым другим элементом управления.

## Пример 3 Код XAML

```
<GroupBox Margin="10" Header="Цвета">
  <StackPanel Orientation="Vertical">
    <Button Background="Red">Красный</Button>
    <Button Background="Green">Зеленый</Button>
    <Button Background="Blue">Синий</Button>
    <Button Background="Yellow">Желтый</Button>
    <Button Background="Brown">Коричневый</Button>
  </StackPanel>
</GroupBox>
```

## Результат



## Элемент управления *Expander*

Данный элемент управления содержит область, которую пользователь может раскрыть или скрыть, кликнув на заголовок. Может содержать только один дочерний элемент (обычно – диспетчер компоновки).

Основные свойства:

**Header** – заголовок элемента управления. Может быть текстовой строкой, либо любым другим элементом управления;

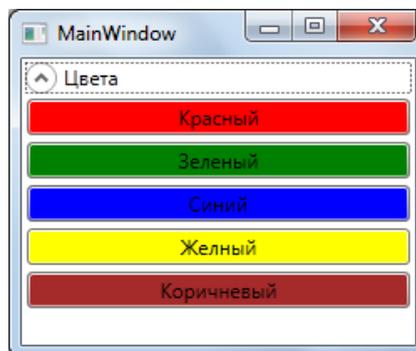
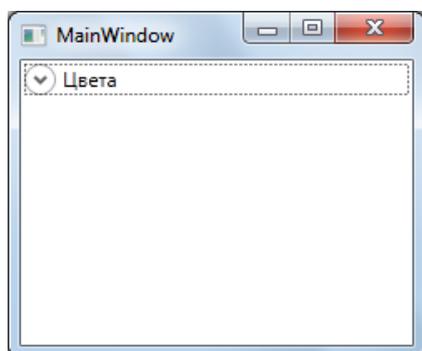
**IsExpanded** – состояние элемента управления: True – раскрыт, False – закрыт;

**ExpandDirection** – направление раскрытия: Down (значение по умолчанию), Up, Left, Right.

### Пример 4 Код XAML

```
<Expander Header="Цвета">
  <StackPanel Orientation="Vertical">
    <Button Background="Red">Красный</Button>
    <Button Background="Green">Зеленый</Button>
    <Button Background="Blue">Синий</Button>
    <Button Background="Yellow">Желтый</Button>
    <Button Background="Brown">Коричневый</Button>
  </StackPanel>
</Expander>
```

### Результат



## Элемент управления *TabItem*

Данный элемент управления представляет собой страницу-вкладку с заголовком в элементе управления вкладками *TabControl*. Переключение между вкладками происходит при нажатии на заголовок вкладки. *TabItem* может содержать только один дочерний элемент (обычно – диспетчер компоновки).

Основные свойства:

**Header** – заголовок элемента управления. Может быть текстовой строкой, либо любым другим элементом управления;

**IsSelected** – состояние вкладки: True – раскрыта, False – скрыта.

### Пример 5 Код XAML

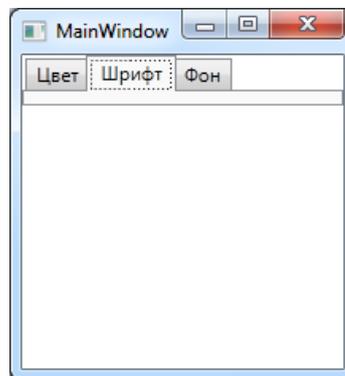
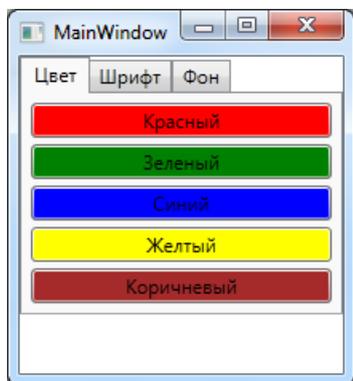
```
<TabControl>
  <TabItem Header="Цвет">
    <StackPanel Orientation="Vertical">
      <Button Background="Red">Красный</Button>
    </StackPanel>
  </TabItem>
</TabControl>
```

```

        <Button Background="Green">Зеленый</Button>
        <Button Background="Blue">Синий</Button>
        <Button Background="Yellow">Желтый</Button>
        <Button Background="Brown">Коричневый</Button>
    </StackPanel>
</TabItem>
<TabItem Header="Шрифт">
</TabItem>
<TabItem Header="Фон">
</TabItem>
</TabControl>

```

## Результат



## Кисти

Кисти в WPF-приложениях используются рисования линий и для заполнения областей: фонов, передних планов, границ элементов, областей прозрачности.

### Кисть *SolidColorBrush*

Используется для рисования линий и заполнения области одним сплошным цветом. Данный тип кисти используется в случаях, когда для свойств `Foreground`, `Background`, `Fill`, `Stroke` заданы текстовые значения в виде названия цвета.

Пример 6 Код XAML	Результат
<pre> &lt;TextBlock&gt;   &lt;TextBlock.Background&gt;     &lt;SolidColorBrush Color="DarkGreen" /&gt;   &lt;/TextBlock.Background&gt;   &lt;TextBlock.Foreground&gt;     &lt;SolidColorBrush Color="White" /&gt;   &lt;/TextBlock.Foreground&gt;   Кисть SolidColorBrush &lt;/TextBlock&gt; </pre>	
<pre> &lt;TextBlock Background="DarkGreen" Foreground="White"&gt;   Кисть SolidColorBrush &lt;/TextBlock&gt; </pre>	

## Кисть RadialGradientBrush

Используется для радиального градиентного заполнения области. Основные свойства:

**GradientStops** – коллекция объектов GradientStop, определяющих промежуточные точки градиента;

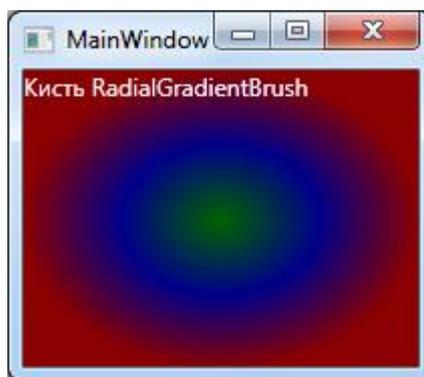
**GradientOrigin** – относительные координаты центра радиального градиента в формате «X,Y».

Значение по умолчанию «0.5,0.5».

### Пример 7 Код XAML

```
<TextBlock Foreground="White" Height="150">
  <TextBlock.Background>
    <RadialGradientBrush>
      <GradientStop Color="DarkGreen" Offset="0" />
      <GradientStop Color="DarkBlue" Offset="0.5" />
      <GradientStop Color="DarkRed" Offset="1" />
    </RadialGradientBrush>
  </TextBlock.Background>
  Кисть RadialGradientBrush
</TextBlock>
```

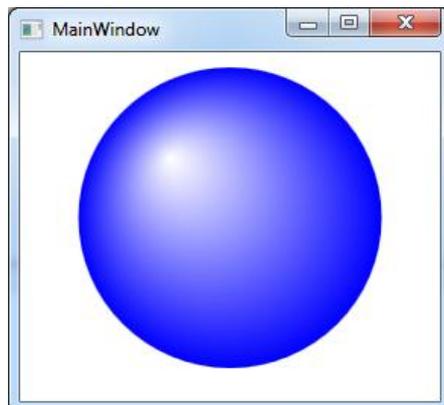
### Результат



### Пример 8 Код XAML

```
<Ellipse Margin="10" Width="200" Height="200">
  <Ellipse.Fill>
    <RadialGradientBrush GradientOrigin="0.3,0.3">
      <GradientStop Color="White" Offset="0"></GradientStop>
      <GradientStop Color="Blue" Offset="1"></GradientStop>
    </RadialGradientBrush>
  </Ellipse.Fill>
</Ellipse>
```

### Результат



## Кисть LinearGradientBrush

Используется для линейного градиентного заполнения области. Основные свойства:

**GradientStops** – коллекция объектов GradientStop, определяющих промежуточные точки градиента;

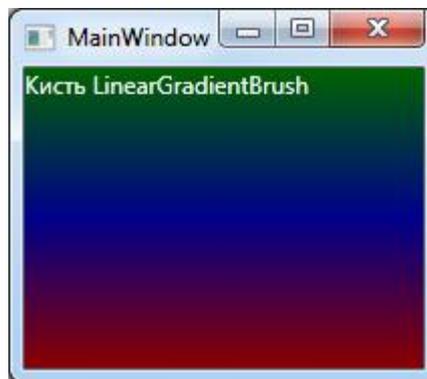
**StartPoint** – относительные координаты начала линейного градиента в формате «X,Y». Значение по умолчанию «0,0»;

**EndPoint** – относительные координаты конца линейного градиента в формате «X,Y». Значение по умолчанию «1,1».

### Пример 9 Код XAML

```
<TextBlock Foreground="White" Height="150">
  <TextBlock.Background>
    <LinearGradientBrush StartPoint="0,0" EndPoint="0,1">
      <GradientStop Color="DarkGreen" Offset="0" />
      <GradientStop Color="DarkBlue" Offset="0.5" />
      <GradientStop Color="DarkRed" Offset="1" />
    </LinearGradientBrush>
  </TextBlock.Background>
  Кисть LinearGradientBrush
</TextBlock>
```

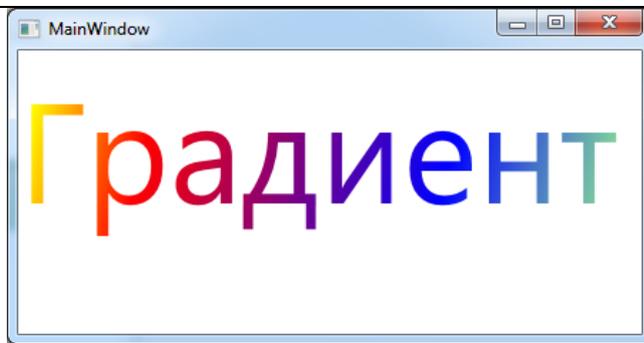
### Результат



### Пример 10 Код XAML

```
<TextBlock FontSize="100">
  <TextBlock.Foreground>
    <LinearGradientBrush>
      <GradientStop Color="Yellow" Offset="0.0" />
      <GradientStop Color="Red" Offset="0.2" />
      <GradientStop Color="Blue" Offset="0.7" />
      <GradientStop Color="LightGreen" Offset="1.0" />
    </LinearGradientBrush>
  </TextBlock.Foreground>
  Градиент
</TextBlock>
```

### Результат



## Кисть *ImageBrush*

Используется для заполнения области графическим изображением, которое может растягиваться, масштабироваться или многократно повторяться. Основные свойства:

**ImageSource** – определяет изображение (ресурс, внешний файл или URL-адрес).

### Пример 11 Код XAML

```
<TextBlock FontWeight="ExtraBold" Height="150">
  <TextBlock.Background>
    <ImageBrush Opacity="0.3" ImageSource="http://aics.ru/img/logo.gif" />
  </TextBlock.Background>
</TextBlock>
```

Кисть *ImageBrush*  
</TextBlock>

### Результат



## Кисть *VisualBrush*

Используется для заполнения области на основе визуального содержимого какого-либо элемента. Кисть может быть использована для уменьшенного изображения какой-либо области окна (в том числе невидимой в данный момент), а также для создания эффекта отражения. Заполненная область автоматически перерисовывается при каждом изменении внешнего вида исходного элемента. Основные свойства:

**Visual** – ссылка на элемент, визуального изображение которого используется. Значение свойства указывается в виде расширения разметки *Binding*, например: `Visual="{Binding ElementName=ButtonRun}"`

В примере 12 кисть *Visual* используется для вывода уменьшенного изображения вкладки в заголовок этой вкладки.

### Пример 12 Код XAML

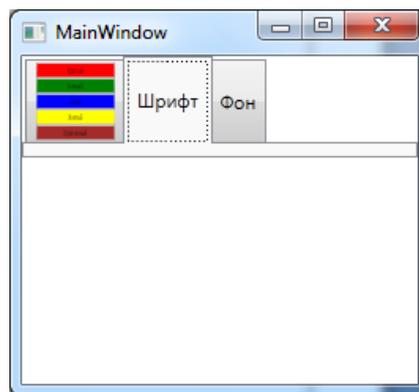
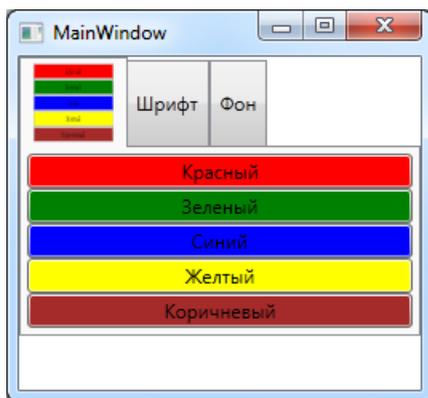
```
<TabControl>
  <TabItem>
```

```

<TabItem.Header>
  <Rectangle Width="50" Height="50">
    <Rectangle.Fill>
      <VisualBrush Visual="{Binding ElementName=colors}" />
    </Rectangle.Fill>
  </Rectangle>
</TabItem.Header>
<StackPanel Orientation="Vertical" x:Name="colors">
  <Button Background="Red">Красный</Button>
  <Button Background="Green">Зеленый</Button>
  <Button Background="Blue">Синий</Button>
  <Button Background="Yellow">Желтый</Button>
  <Button Background="Brown">Коричневый</Button>
</StackPanel>
</TabItem>
<TabItem Header="Шрифт">
</TabItem>
<TabItem Header="Фон">
</TabItem>
</TabControl>

```

## Результат



В примере 13 кисть Visual используется для создания эффекта отражения текстового поля. При изменении текста в текстовом поле отражение автоматически изменяется.

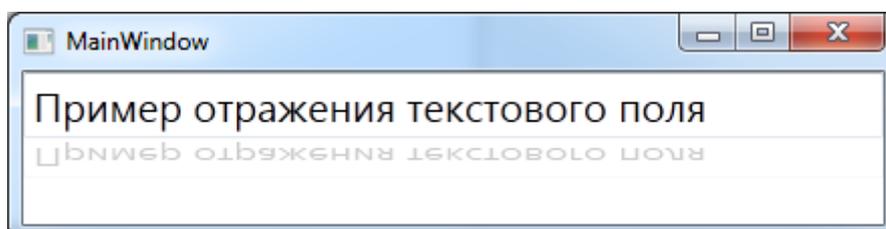
## Пример 13 Код XAML

```

<StackPanel>
  <TextBox Name="txt" FontSize="20">Пример отражения текстового поля</TextBox>
  <Rectangle Height="20" Opacity="0.2" RenderTransformOrigin="0,0.5">
    <Rectangle.Fill>
      <VisualBrush Visual="{Binding ElementName=txt}" />
    </Rectangle.Fill>
    <Rectangle.RenderTransform>
      <ScaleTransform ScaleY="-1"></ScaleTransform>
    </Rectangle.RenderTransform>
  </Rectangle>
</StackPanel>

```

## Результат



### **Задание 1**

На основе примера 8 разработайте WPF-приложение с анимацией источника света, светового пятна от него на шаре и отражения шара. Для анимации начала радиального градиента используйте элемент `PointAnimation`, свойства `From` и `To` которого задаются в формате "X,Y" (пример: `To="0,1"`)