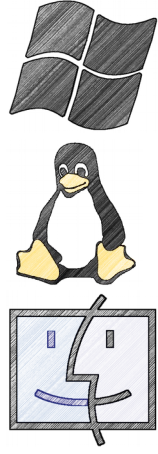
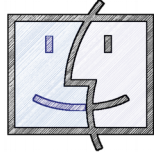


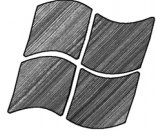
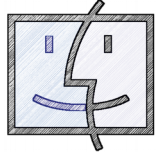
AvaloniaUI

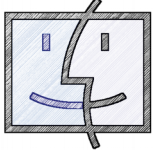
Cross-platform
Open source
UI Framework

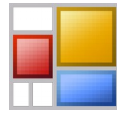
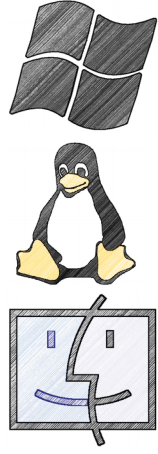
With .NET Core!


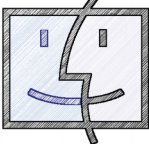
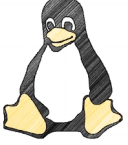















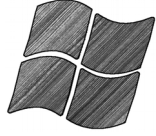
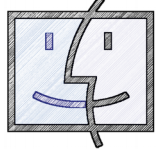


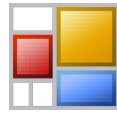
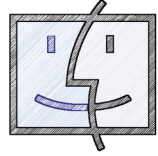
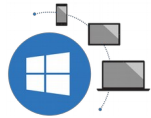






															
---	---	---	---	--	---	--	---	--	---	---	---	---	--	--	---



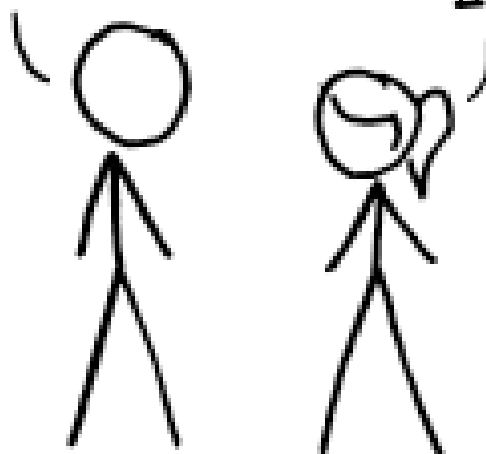


КАК МНОЖАТСЯ СТАНДАРТЫ:

(СМ.: ЗАРЯДНЫЕ УСТРОЙСТВА, КОДИРОВКИ, МГНОВЕННЫЕ СООБЩЕНИЯ И Т.Д.)

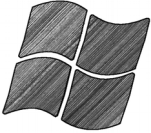






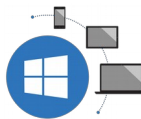




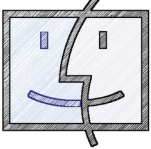








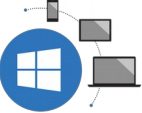


СИТУАЦИЯ:
ЕСТЬ 14
КОНКУРИРУЮЩИХ
СТАНДАРТОВ.







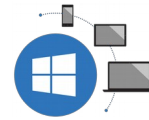


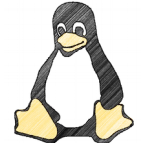
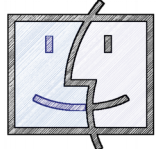


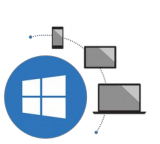
14?! АБСУРД! НАМ
НЕОБХОДИМО
РАЗРАБОТАТЬ ОДИН
УНИВЕРСАЛЬНЫЙ
СТАНДАРТ НА ВСЕ СЛУЧАИ
ЖИЗНИ.

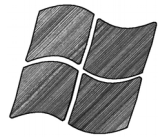


СКОРО

СИТУАЦИЯ:
ЕСТЬ 15
КОНКУРИРУЮЩИХ
СТАНДАРТОВ.

								
	✓	✓	✓					
		✓	✓					
		✓	✓	✓				✓
					✓			✓
						✓		✓
							✓	✓



В чём проблемы обёрток над нативными
контролами?

В чём проблемы обёрток над нативными
контролами?

В чём проблемы обёрток над нативными контролами?

- Приведение к платформ общему знаменателю

В чём проблемы обёрток над нативными контролами?

- Приведение к платформ общему знаменателю
- Сложность разработки UI с богатой стилизацией по макету от дизайнера

Чем WPF/UWP лучше остальных?

Чем WPF/UWP лучше остальных?


- Система привязок
(обходится XForms, Eto.Forms, MvvmCross)

Чем WPF/UWP лучше остальных?

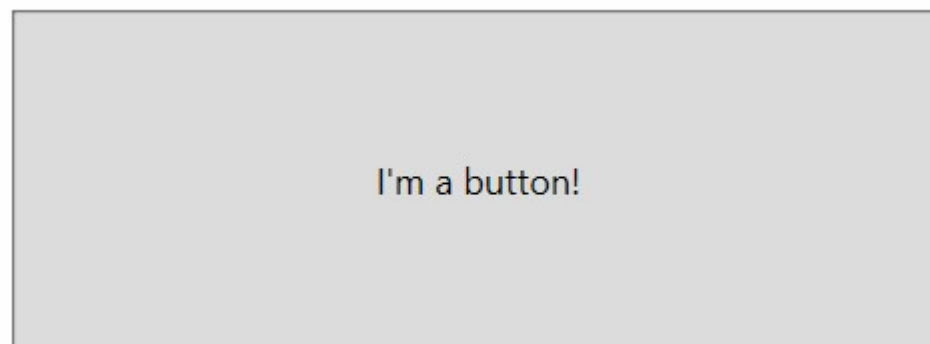
- Система привязок
(обходится XForms, Eto.Forms, MvvmCross)
- Человекочитаемая разметка, в которой можно верстать
(обходится XForms/Eto.Forms, так же нечто похожее есть в Android)






Чем WPF/UWP лучше остальных?

- Система привязок
(обходится XForms, Eto.Forms, MvvmCross)
- Человекочитаемая разметка, в которой можно верстать
(обходится XForms/Eto.Forms, так же нечто похожее есть в Android)
- Структура контролов на шаблонах



I'm a button!



- △  [Button] 
- △  border [Border]
- △  contentPresenter [ContentPresenter]
-  [TextBlock]

<Border

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}"

Background="{TemplateBinding Background}" ...>

<ContentPresenter

ContentTemplate="{TemplateBinding ContentTemplate}"

Content="{TemplateBinding Content}" .../>

</Border>

```
<ControlTemplate.Triggers>
  <Trigger Property="IsMouseOver" Value="True">
    <Setter Property="Background" TargetName="border" Value="#FFBEE6">
    <Setter Property="BorderBrush" TargetName="border" Value="#FF3C7F">
  </Trigger>
  <Trigger Property="IsPressed" Value="True">
    <Setter Property="Background" TargetName="border" Value="#FFC4E5">
    <Setter Property="BorderBrush" TargetName="border" Value="#FF2C62">
  </Trigger>
  <Trigger Property="IsEnabled" Value="False">
    <Setter Property="Background" TargetName="border" Value="#FFF4F4">
    <Setter Property="BorderBrush" TargetName="border" Value="#FFADB2">
    <Setter Property="TextElement.Foreground" TargetName="contentPres
  </Trigger>
</ControlTemplate.Triggers>
```

Преимущества контролов без внешнего вида (lookless)

- Отделение логики работы и свойств контроля от его представления
- Возможность менять вид контроля практически *любым* образом по месту использования

ДЕМО

Чего мы хотим от UI-фреймворка?

- Все плюшки WPF
- Переносимость
- Open-source
- 100% managed-код
- Рендеринг «пиксель-в-пиксель» *везде*
- Использование новых фич языка и экосистемы

AvaloniaUI

(netstandard1.1)

- биндинги
- logical/visual tree
- рендерер(ы)
- контролы
- прочее

AvaloniaUI

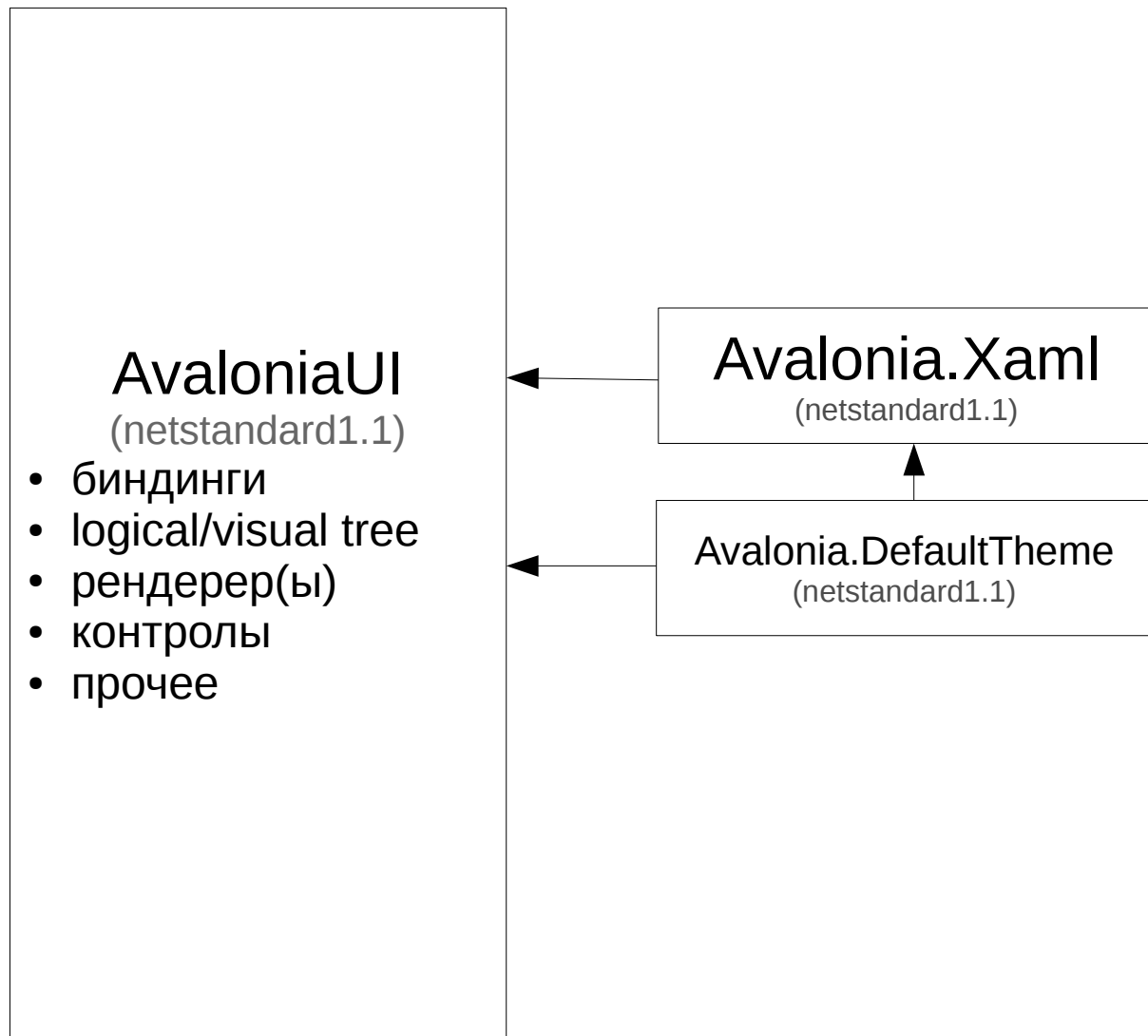
(netstandard1.1)

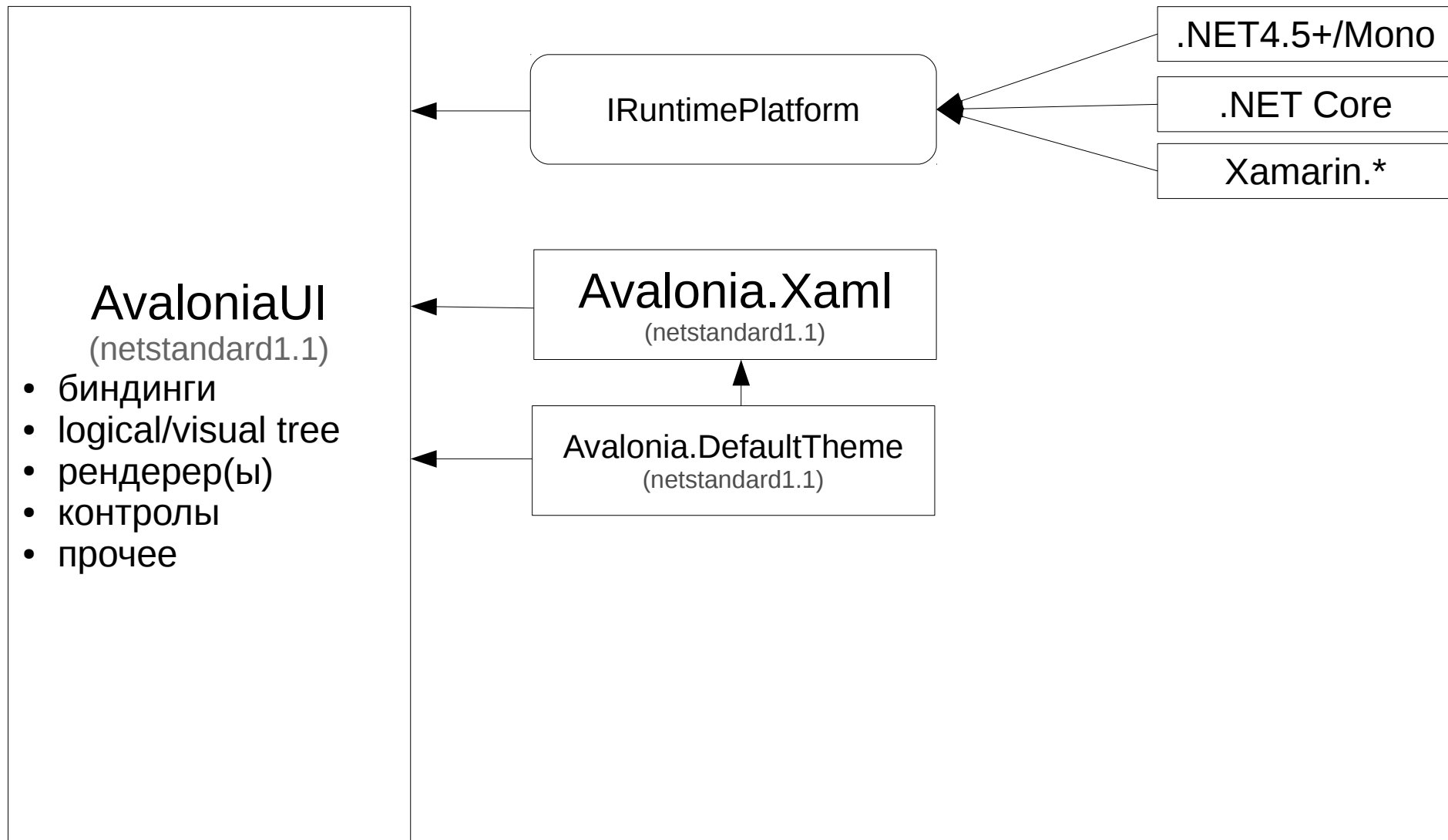
- биндинги
- logical/visual tree
- рендерер(ы)
- контролы
- прочее

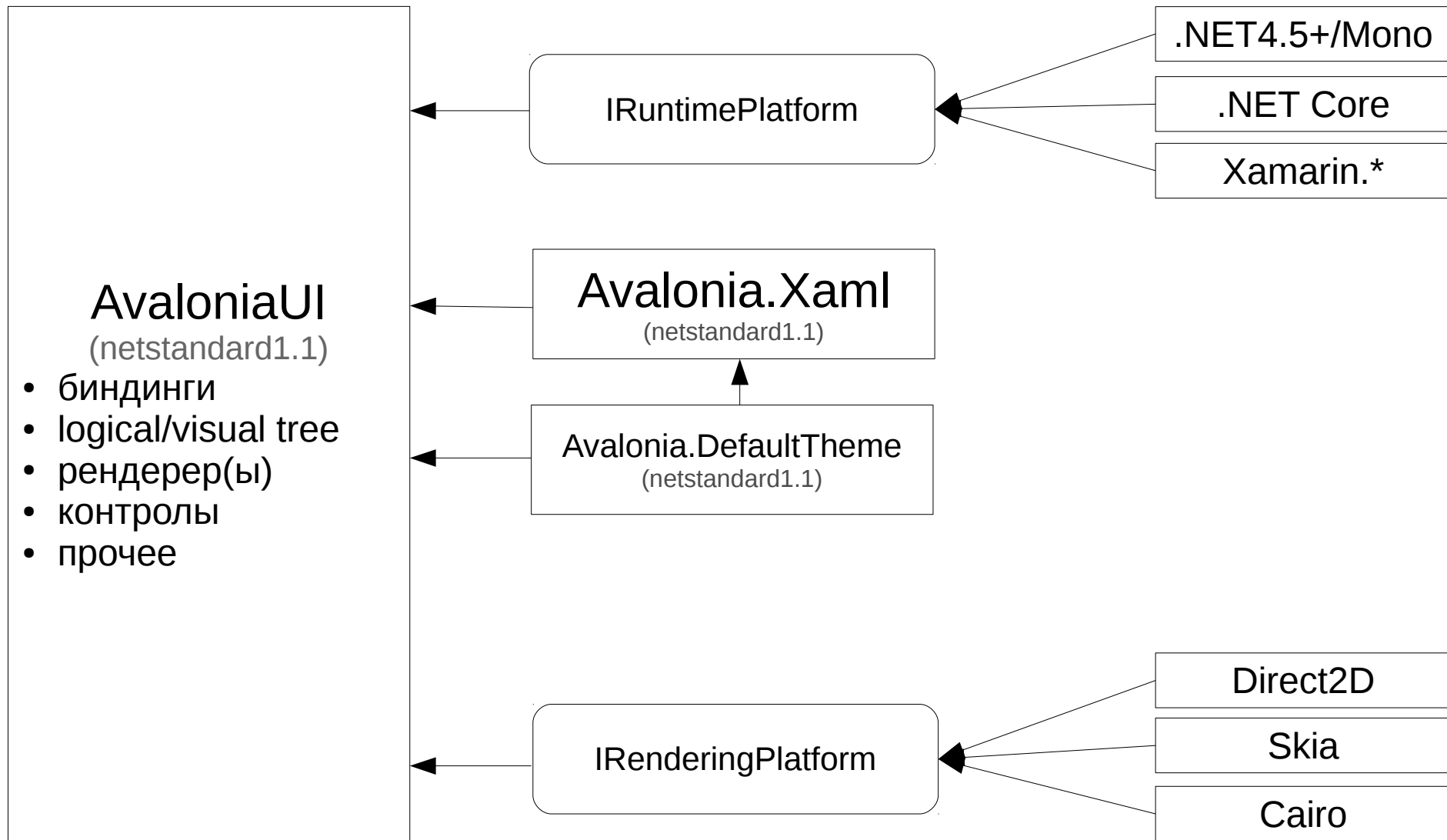
Avalonia.Xaml

(netstandard1.1)



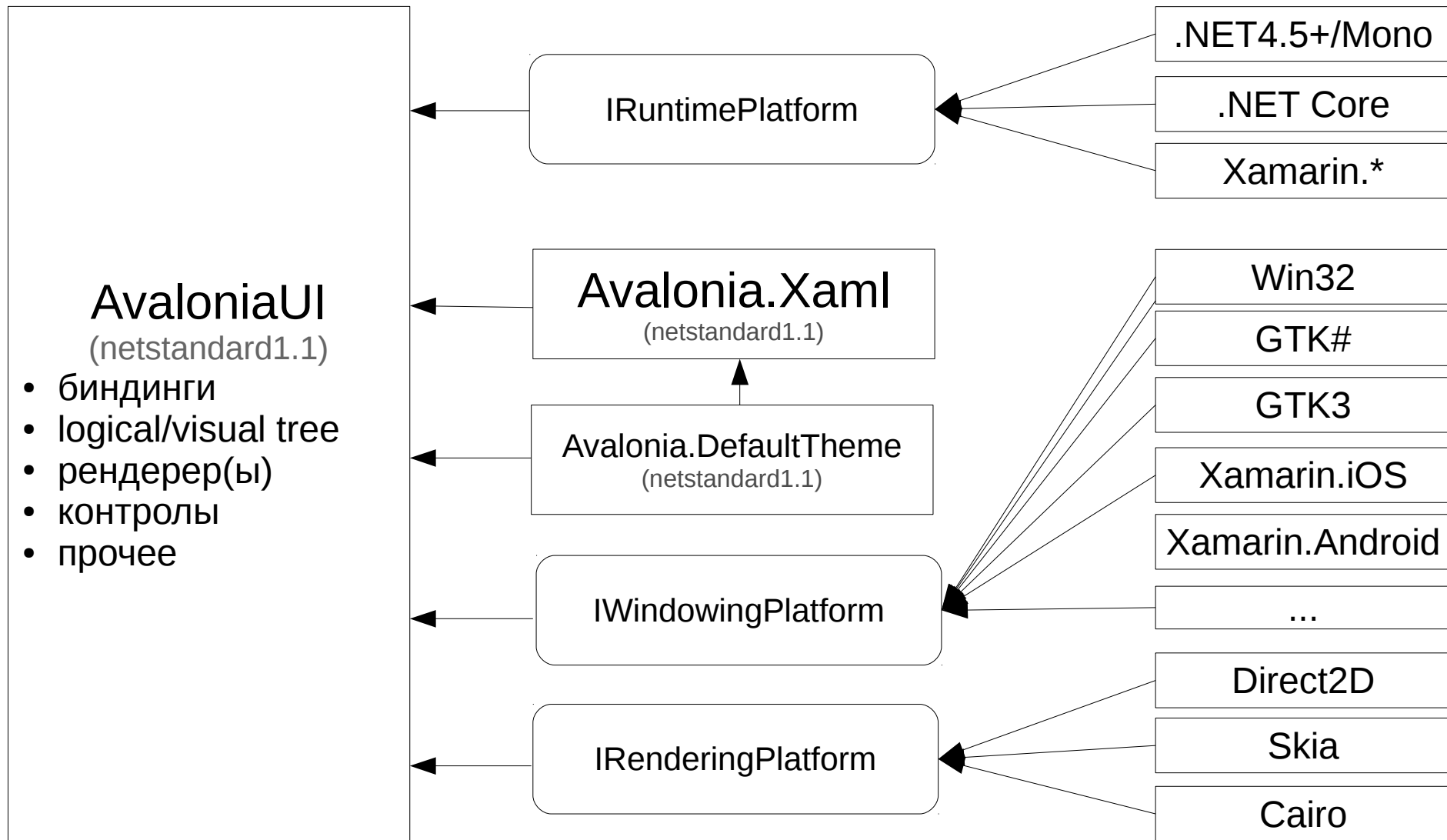






IDrawingContextImpl

```
Matrix Transform { get; set; }  
void Clear(Color color);  
void DrawImage(IBitmapImpl src, double opacity, Rect sect, Rect dest);  
void DrawLine(Pen pen, Point p1, Point p2);  
void DrawGeometry(IBrush brush, Pen pen, IGeometryImpl geometry);  
void DrawRectangle(Pen pen, Rect rect, float cornerRadius);  
void DrawText(IBrush color, Point origin, IFormattedTextImpl text);  
void FillRectangle(IBrush brush, Rect rect, float cornerRadius);  
void PushClip(Rect clip);  
void PushOpacity(double opacity);
```

ITopLevelImpl

```
Size ClientSize { get; }
double Scaling { get; }
IEnumerable<object> Surfaces { get; }
Action<RawInputEventArgs> Input { get; set; }
Action<Rect> Paint { get; set; }
Action<Size> Resized { get; set; }
Action<double> ScalingChanged { get; set; }
void Invalidate(Rect rect);
void SetInputRoot(IInputRoot inputRoot);
Point PointToClient(Point point);
Point PointToScreen(Point point);
void SetCursor(IPlatformHandle cursor);
Action Closed { get; set; }
```

ITopLevelImpl

```

Size ClientSize { get; }
double Scaling { get; }
IEnumerable<object> Surfaces { get; }
Action<RawInputEventArgs> Input { get; set; }
Action<Rect> Paint { get; set; }
Action<Size> Resized { get; set; }
Action<double> ScalingChanged { get; set; }
void Invalidate(Rect rect);
void SetInputRoot(IInputRoot inputRoot);
Point PointToClient(Point point);
Point PointToScreen(Point point);
void SetCursor(IPlatformHandle cursor);
Action Closed { get; set; }

```

Иерархия виджетов верхнего уровня

TopLevel

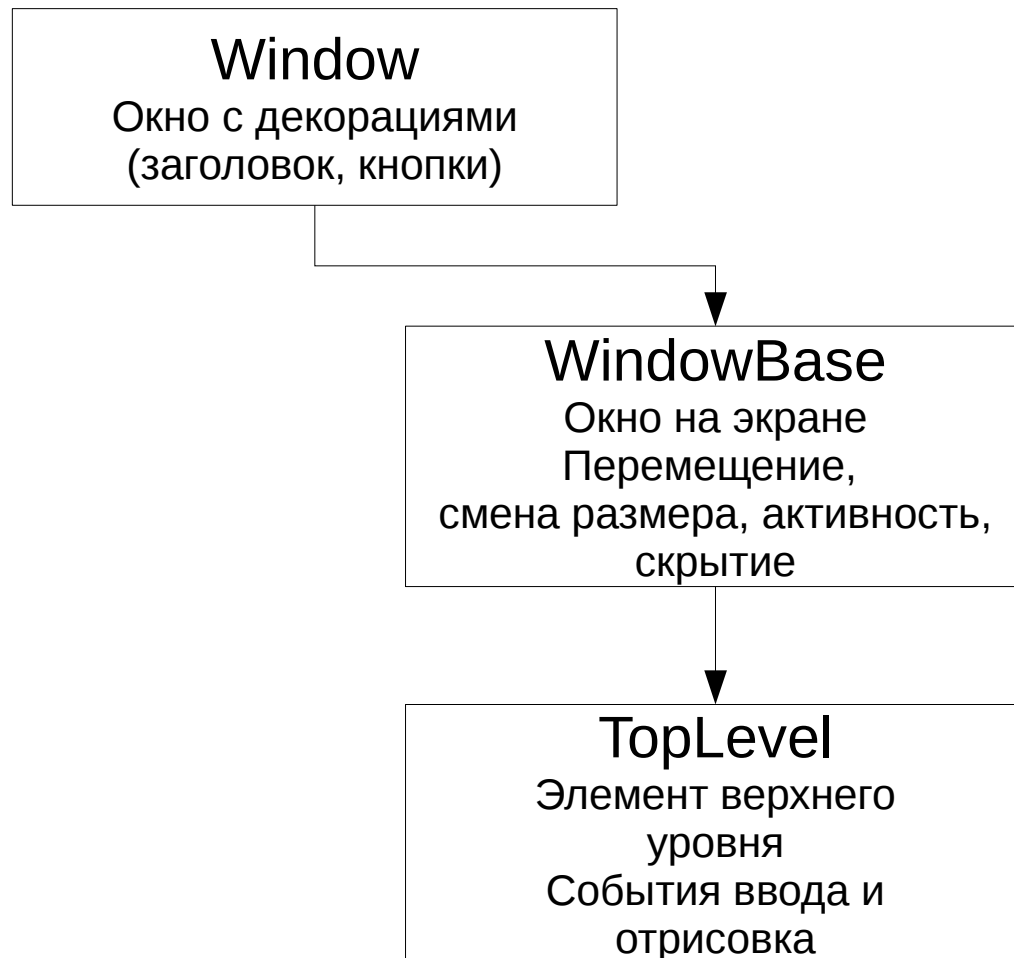
Элемент верхнего
уровня

События ввода и
отрисовка

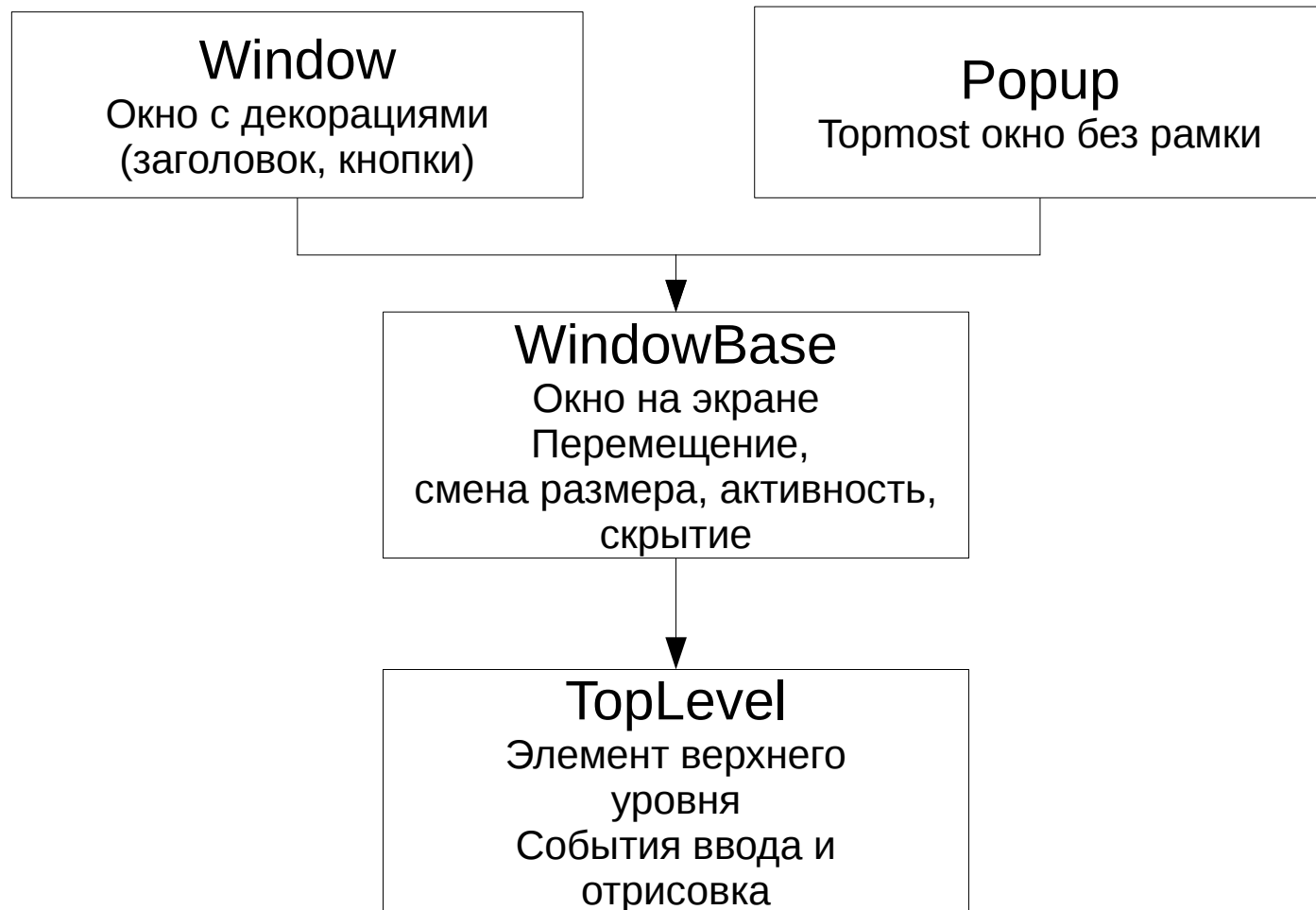
Иерархия виджетов верхнего уровня



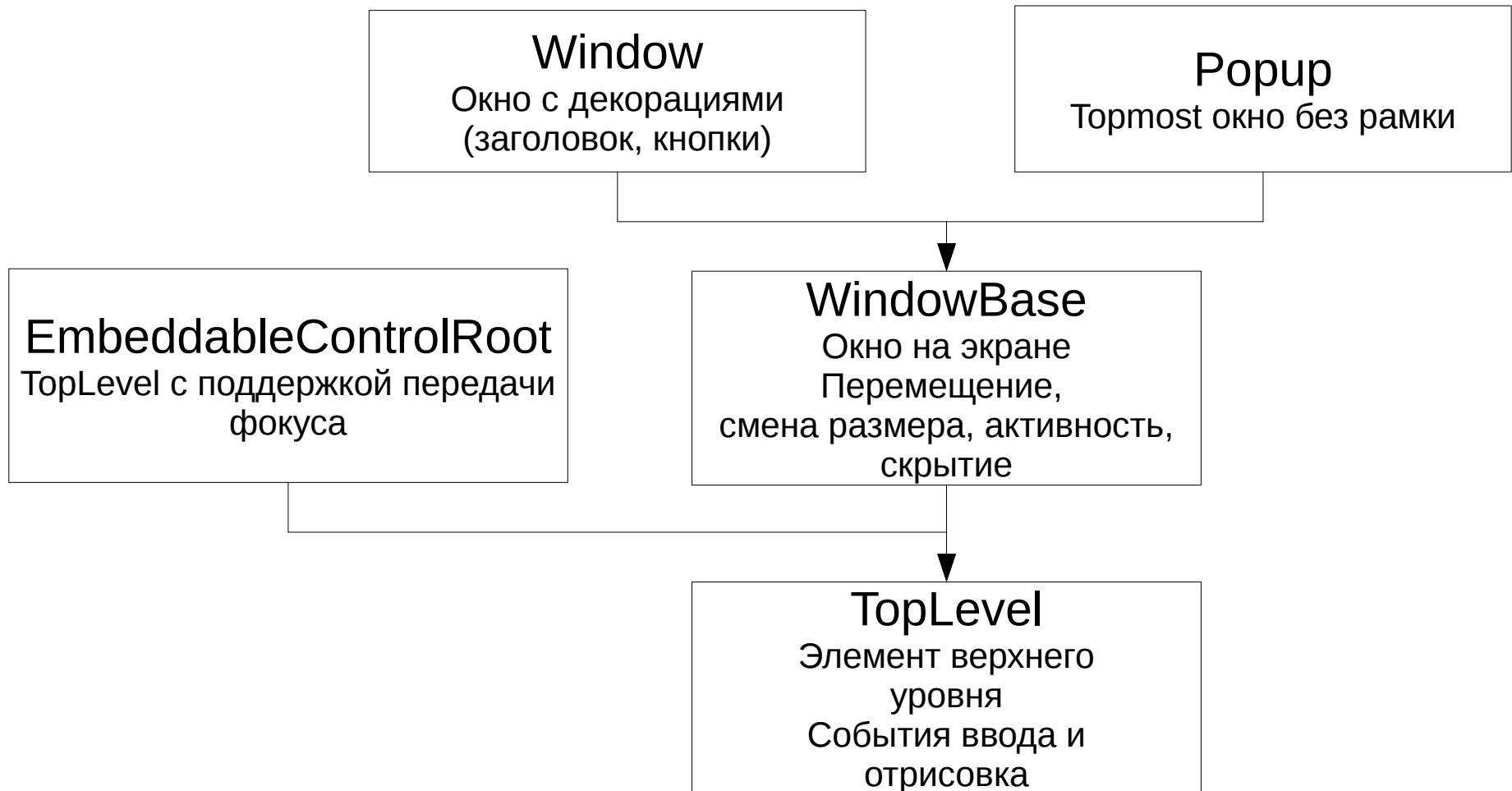
Иерархия виджетов верхнего уровня



Иерархия виджетов верхнего уровня



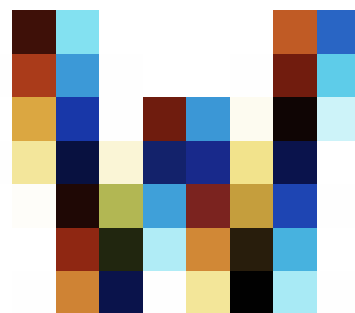
Иерархия виджетов верхнего уровня



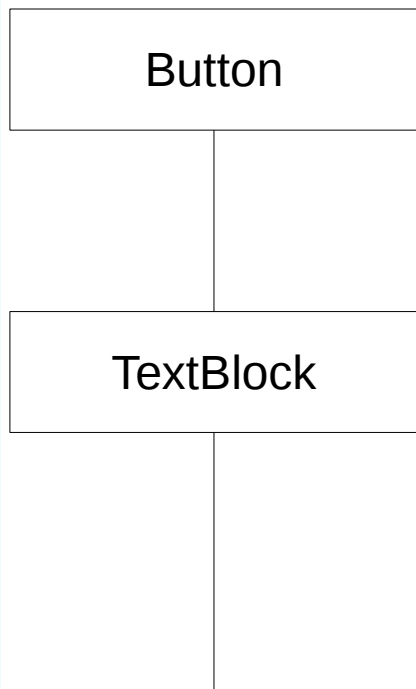
ДЕМО

Проблемы наивного метода отрисовки (рекурсивно бежим по сцене)

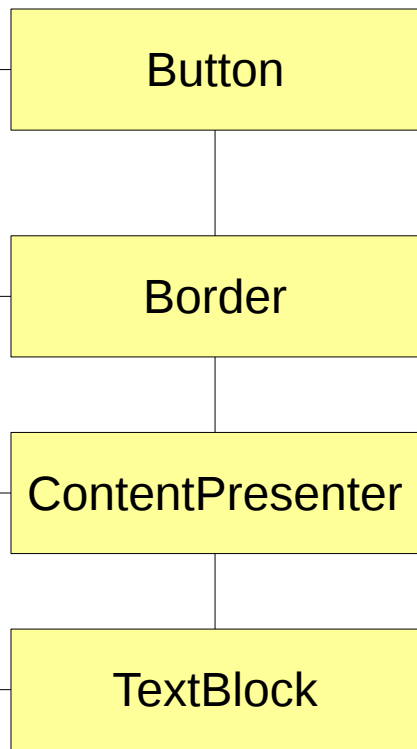
- Сложно понять, что надо отрисовывать, а что нет
- Занимаем много времени на UI-потоке
- Полупрозрачная отрисовка тормозит
- Отрисовка с поворотами и масштабированием тормозит (особенно шрифты)



Logical Tree

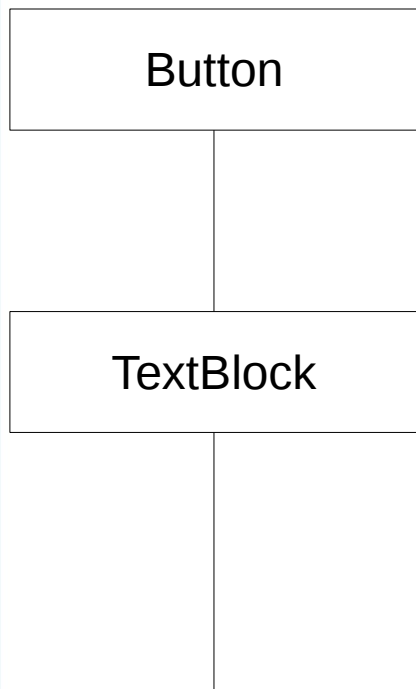


Visual Tree

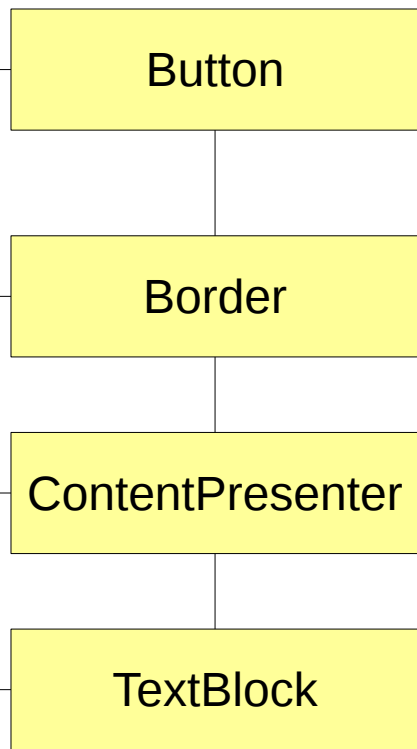


Render Tree

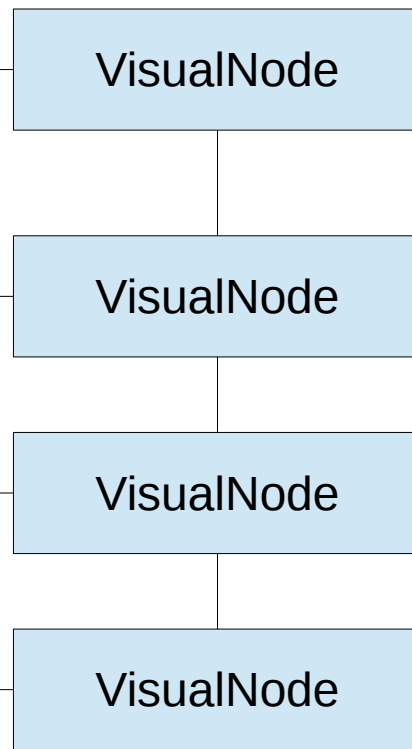
Logical Tree



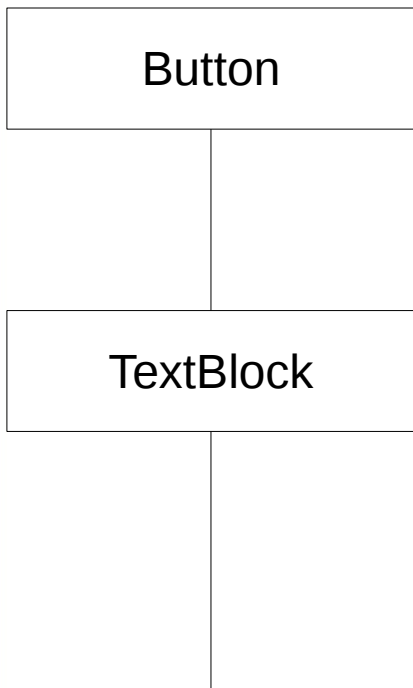
Visual Tree



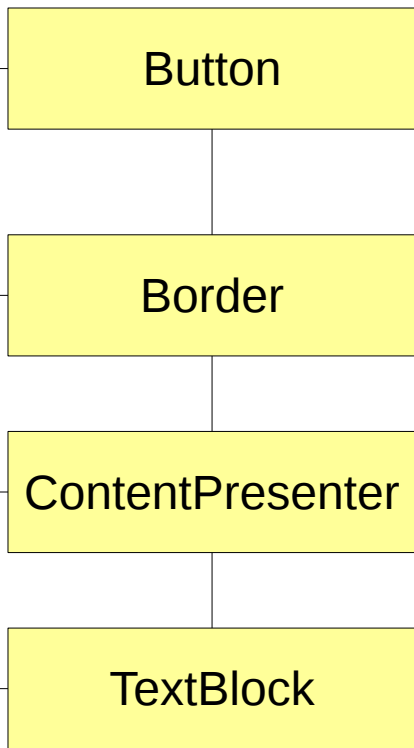
Render Tree



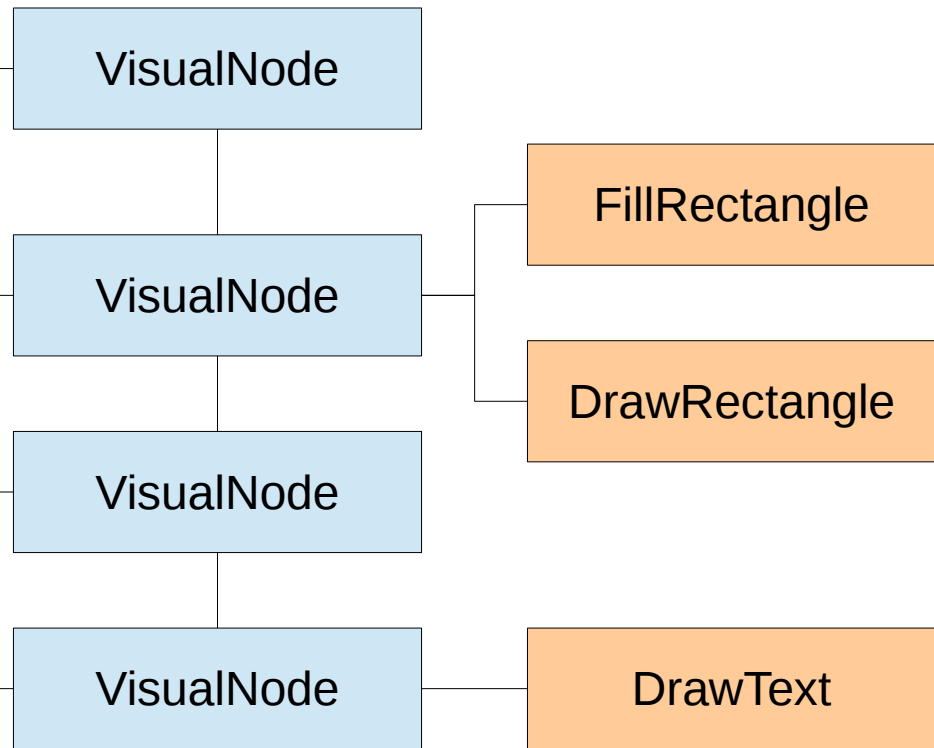
Logical Tree



Visual Tree



Render Tree



ДЕМО

Отличия от WPF

- AvaloniaObject (то же самое, что и DependencyObject)
- Использует ReactiveExtensions
- Другая система стилей
- Отличия в иерархии контролов
- AvaloniaProperty
(как DependencyProperty, но строго типизирована)
 - StyledProperty
 - DirectProperty

```
public static readonly DependencyProperty<TextBox, int>
    SelectionEndProperty =
    AvaloniaProperty.RegisterDirect<TextBox, int>(
        nameof(SelectionEnd),
        o => o.SelectionEnd,
        (o, v) => o.SelectionEnd = v);
```

```
public static readonly DependencyProperty<TextBox, int>
    SelectionEndProperty =
    AvaloniaProperty.RegisterDirect<TextBox, int>(
        nameof(SelectionEnd),
        o => o.SelectionEnd,
        (o, v) => o.SelectionEnd = v);

public int SelectionEnd
{
    get { return _selectionEnd; }
    set { SetAndRaise(SelectionEndProperty, ref
        _selectionEnd, CoerceCaretValue(value)); }
}
```

ДЕМО

Чего пока нет?

(версия 0.5)

- Стабильности внутреннего API
- Не хватает многих подсистем, таких как API работы со шрифтами и обработки тач-событий
- Поддержка мобильных устройств на экспериментальном уровне
- «Умный» рендерер пока не в главной ветке
- Для деплоя требует знания целевых платформ
- Документация :(
- Интеграция с IDE
- Выловлены не все проблемы с особенностями платформ


```
SetDefaultSize(200, 200);  
btn.Clicked += delegate  
{  
    int x, y;  
    this.GetSize(out x, out y);  
    Console.WriteLine("Clicked ClientSize: " +  
        x + " " + y);  
  
    Resize(x + 20, y + 30);  
  
    ///???  
    this.GetSize(out x, out y);  
    Console.WriteLine("AfterClicked ClientSize: " +  
        x + " " + y);  
};
```

```
SetDefaultSize(200, 200);
```

```
btn.Clicked += delegate
```

```
{
```

```
    int x, y;
```

```
    this.GetSize(out x, out y);
```

```
    Console.WriteLine("Clicked ClientSize: " +  
        x + " " + y);
```

```
    Resize(x + 20, y + 30);
```

```
    ///???
```

```
    this.GetSize(out x, out y);
```

```
    Console.WriteLine("AfterClicked ClientSize: " +  
        x + " " + y);
```

```
};
```

```
SetDefaultSize(200, 200);  
btn.Clicked += delegate  
{
```

```
    int x, y;  
    this.GetSize(out x, out y);  
    Console.WriteLine("Clicked ClientSize: " +  
        x + " " + y);
```

```
    Resize(x + 20, y + 30);
```

```
    ///???
```

```
    this.GetSize(out x, out y);  
    Console.WriteLine("AfterClicked ClientSize: " +  
        x + " " + y);
```

```
};
```

```
SetDefaultSize(200, 200);  
btn.Clicked += delegate  
{  
    int x, y;  
    this.GetSize(out x, out y);  
    Console.WriteLine("Clicked ClientSize: " +  
        x + " " + y);  
  
    Resize(x + 20, y + 30);  
  
    ///???  
    this.GetSize(out x, out y);  
    Console.WriteLine("AfterClicked ClientSize: " +  
        x + " " + y);  
};
```

```
SetDefaultSize(200, 200);  
btn.Clicked += delegate  
{  
    int x, y;  
    this.GetSize(out x, out y);  
    Console.WriteLine("Clicked ClientSize: " +  
        x + " " + y);  
  
    Resize(x + 20, y + 30);
```

```
    ///???  
    this.GetSize(out x, out y);  
    Console.WriteLine("AfterClicked ClientSize: " +  
        x + " " + y);  
};
```

a) 220, 230

б) 230, 220

в) 220, 268

г) 180, 170

а) 220, 230

б) 230, 220

в) 220, 268

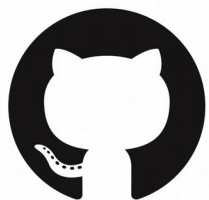
г) 180, 170

д) правильного ответа
среди вариантов нет

Запланировано на 0.6

- Генерация бандлов для OSX и пакетов для линукса в один клик
- Переезд на Portable.Xaml
- Нативный бэкэнд для OSX
- Сделать превьювер для *nix-платформ и интегрировать его хоть куда-нибудь
- Доделать рендерер в отдельном потоке
- Доработать документацию

AvaloniaUI.github.io



<https://github.com/AvaloniaUI/Avalonia>



<https://gitter.im/AvaloniaUI/Avalonia>

Спикер

Никита Цуканов

nikita.d.tsukanov@gmail.com Skype: kekekeks