РАЗРАБОТКА ПРИЛОЖЕНИЙ НА ПЛАТФОРМЕ .NET

Лекция 1

Основы Windows Presentation Foundation
XAML

Обработка графики

- Графика в Windows Forms
  - Рисование силами ОС (использование Windows API)
  - User32 – внешний вид окон, элементов управления (кнопок, текстовые поля и т.д.)
  - GDI / GDI+ - рисование фигур, текста, изображений.
  - Обработка графики Центральным процессором (CPU)
- Графика в WPF
  - Использование DirectX. Отображением элементов занимается DirectX.
  - Ускорение за счет аппаратных средств графической подсистемы (GPU).
  - Совсем не используется системный модуль GDI / GDI+.
  - User32 – по прежнему используется, но в минимальных количествах. Обработка и маршрутизация ввода, определение участка экрана, принадлежащего приложению.
- Следствия использования DirectX в WPF
  - Более богатые графические возможности
  - Поддержка 3D графики
  - Поддержка произвольной анимации и мультимедиа
  - При проектирование интерфейса WPF обычно используется векторная графика

Единицы измерения

- В Windows Forms
  - пиксель
- В WPF
  - независимая от разрешения единица измерения (равная 1/96 дюйма)
  - Элементы выгладят одинаково на экранах с разным разрешением. На экранах с более высоким DPI (точек на дюйм экрана) интерфейс более четко прорисовывается, а не уменьшается в размерах
  - Используется системная установка DPI
  - Размер элемента пересчитывается на текущее значение DPI в системе.
  - При дробном значении пикселя используется сглаживание.

Возможность WPF

- Пример:

- В Windows Forms разные кнопки
  - Кнопка с текстом – просто
  - Кнопка с текстом и рисунком – сложнее
  - Кнопка с видео – неподъемно...

- В WPF и XAML все 3 кнопки делаются одинаково просто. Это возможно благодаря возможности вкладывать одни элементы управления в другие
  - Это применимо не только к кнопкам, но и почти ко всем другим элементам управления

Особенности WPF

- Аппаратное ускорение
- Независимость от разрешения
- Отсутствие фиксированного внешнего вида
- Декларативное описание пользовательского интерфейса (XAML)
  - XAML - декларативный язык описания интерфейса
- Рисование на основе объектов
- Поддержка аудио и видео
- Продвинутые возможности отображения текстовых документов

- Анимация. Декларативное описание анимации
- Система команд
- Поддержка стилей, тем и шаблонов
- Привязки (Binding)

Идеология WPF

- Разделение логики и оформления
  - Бизнес логика – C#
  - Оформление (интерфейс) – XAML
- Разделение задач
  - Логика – программист
  - Интерфейс – дизайнер (используя, например, Expression Blend)
- Подходы для создания пользовательского интерфейса
  - Декларативный (XAML)
  - Императивный подходы (C#)
- Независимость от разрешения экрана
  - Произвольное изменение размеров окон
  - Автоматическая адаптация под содержимое (например, локализованные ресурсы)
  - Гибкая компоновка пользовательского интерфейса

XAML

eXtended Application Markup Language

расширенный язык разметки приложений

Что такое XAML

- XAML – eXtended Application Markup Language – расширенный язык разметки приложений
- Основан на XML. Расширяет его
- Это декларативный язык, описывающий структуру графического интерфейса, стили и сценарии
- Декларативность – описание структуры и свойств, без кода

Особенности WPF

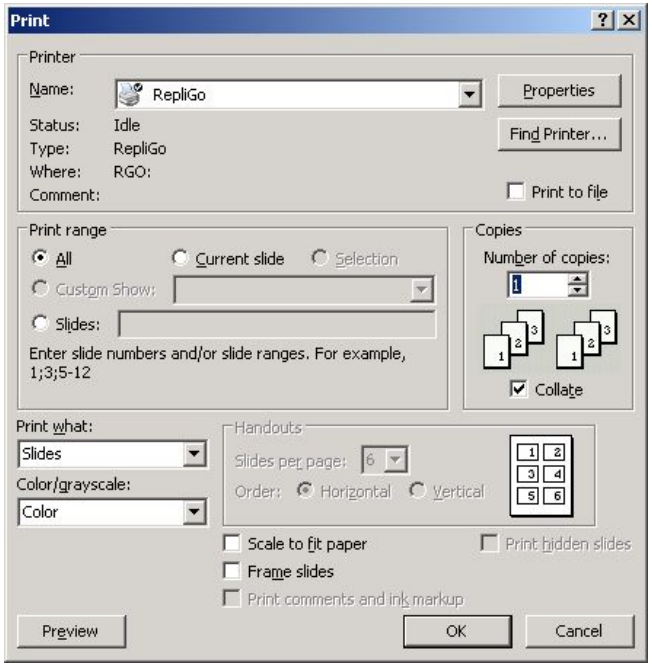
- Использование XAML для
  - Определения структуры (типа HTML)
  - Задания стилей (типа CSS)
  - Анимации и мультимедиа
  - Трехмерной графики и анимации
- Использование C# для
  - Обработки событий
  - Логики приложения

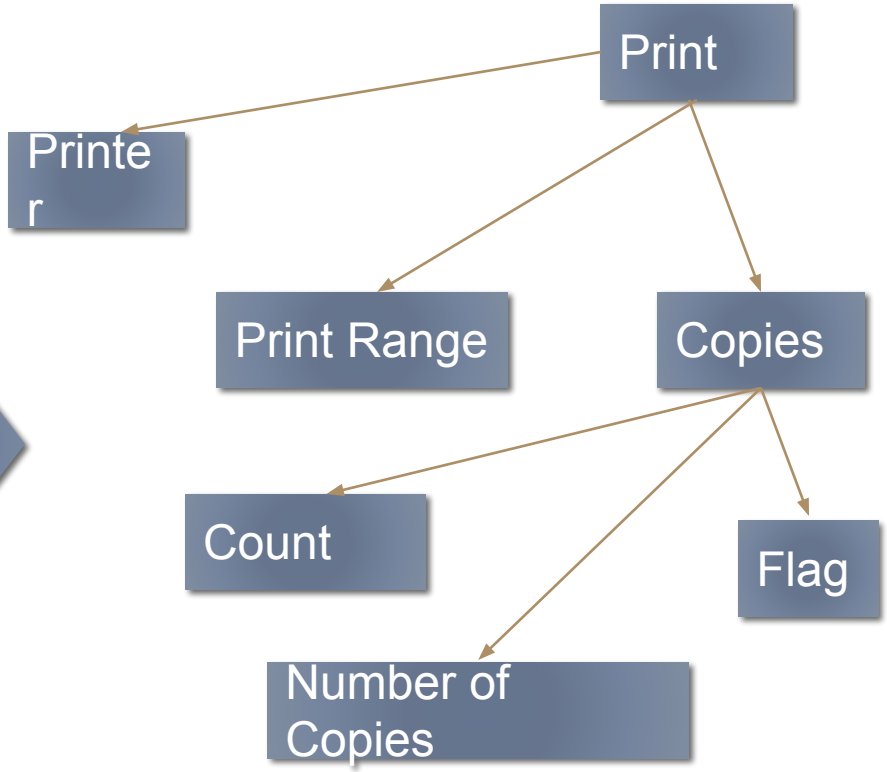
Применение XAML

- Применение XAML
  - Windows Presentation Foundation (WPF)
  - Silverlight
  - XAML может использоваться в любой другой предметной области для декларативного описания, используя пользовательское множество объектов
- WPF использует XAML, но может и обойтись без него
- XAML используется и в других областях, не только в WPF

Структура интерфейса

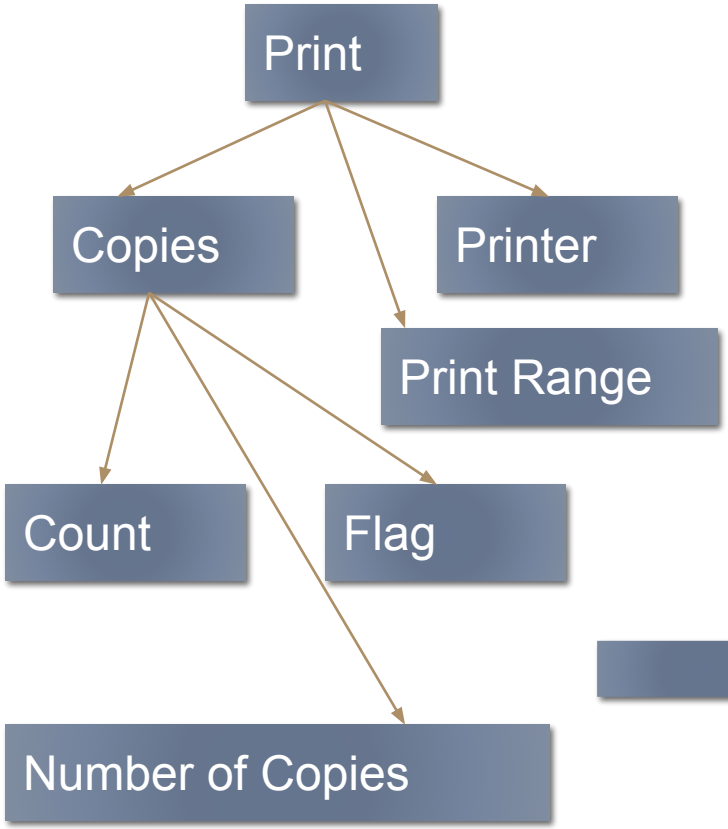
очень похожа на дерево...





Для описания деревьев

- Используют XML – eXtended Markup Language




```
<Window Title="Print">

  <GroupBox Title="Copies">
    <CheckBox Title="Collate"
IsChecked="true"/>
    <Label Title="Number of Copies">
    <Spinner MinValue=1 MaxValue=1000 />
  </GroupBox>

  <GroupBox Title="Print">
  </GroupBox>
</Window>
```

Элементы UI

- Элементам UI соответствуют
  - С одной стороны – элементы XAML
  - С другой стороны – классы .NET
- Имеется соответствие между .NET и XAML

Соответствие XAML□□ C#

XAML	C#
Элемент	Объект класса
Атрибут	Свойство
Вложенность	Спец. свойство
-	Метод
Триггер	-
Сеттер	-
Атрибут-событие	Подписка на событие

Пространства имен XAML

- xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" – основное пространство имен WPF. Охватывает все классы WPF, включая все классы элементов управления. По умолчанию задается как основное пространство имен в XAML.

- xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" – пространство имен XAML. Содержит служебные свойства XAML, которые влияют на то как интерпретируется документ.

- Можно добавлять и другие пространства имен для использования в XAML.
- Синтаксис:
  - xmlns:*Префикс*="clr-namespace:*ПространствоИмен*;assembly=*ИмяСборки*"
    - *Префикс* – префикс XML, который будет использован для указания пространства имен в разметке XAML
    - *ПространствоИмен* – полное название пространства имен C#
    - *ИмяСборки*– имя сборки, в которой описано пространство имен (без dll).
  - xmlns:Sys="clr-namespace:System;assembly=mscorlib"
  - xmlns:Col="clr-namespace:System.Collections.Generic;assembly=mscorlib"

Атрибуты

- <TextBlock Text="Привет"/> - все понятно: Text – типа string

- <TextBlock Margin="2,3,3,1"/> - Свойство Margin имеет тип Thickness. Непонятно как по строке создать тип Thickness и задать ему параметры.

- Для этого анализатор XAML использует конвертеры типов. Класс – конвертер указывается с помощью атрибута TypeConverter для свойства (например, Margin) или для класса (например, Thickness). Анализатор XAML ищет конвертер, преобразует с его помощью строку в нужные тип и присваивает результат свойству.

Присоединенные атрибуты

- Свойства, которые определены в одном классе, а применяются во многих других классах, не связанных наследованием с определяющим классом.

- Синтаксис:
  - ОпределяемыйТип.ИмяСвойства=“значение”

```
<Grid>
    <ComboBox Grid.Row="0" …./>
</Grid>
```

Именование элементов

- Свойства Name и x:Name

- При задании имени в автоматически сгенерированной части класса создается поле с таким именем и типом соответствующим типу элемента.

- В отличии от Windows Forms, элемент в WPF не обязан иметь имя. Имя необходимо задавать, если элемент предполагается использовать в коде или необходимо ссылаться на элемент в XAML

Вложенные элементы

- Каждый элемент сам решает, как поступать со своими вложенными элементами
  - Если родительский элемент реализует интерфейс IList, анализатор XAML вызывает метод IList.Add(), передавая вложенный элемент.
  - Если родительский элемент реализует интерфейс IDictionary, анализатор XAML вызывает метод IDictionary.Add(), передавая вложенный элемент. При этом необходимо задать свойство x:Key для каждого вложенного элемента.

```
    <Button>Текст кнопки</Button>
<ListBox>
    <CheckBox Content="Red"/>
    <CheckBox>Green</CheckBox>
    <CheckBox>Blue</CheckBox>
</ListBox>
```