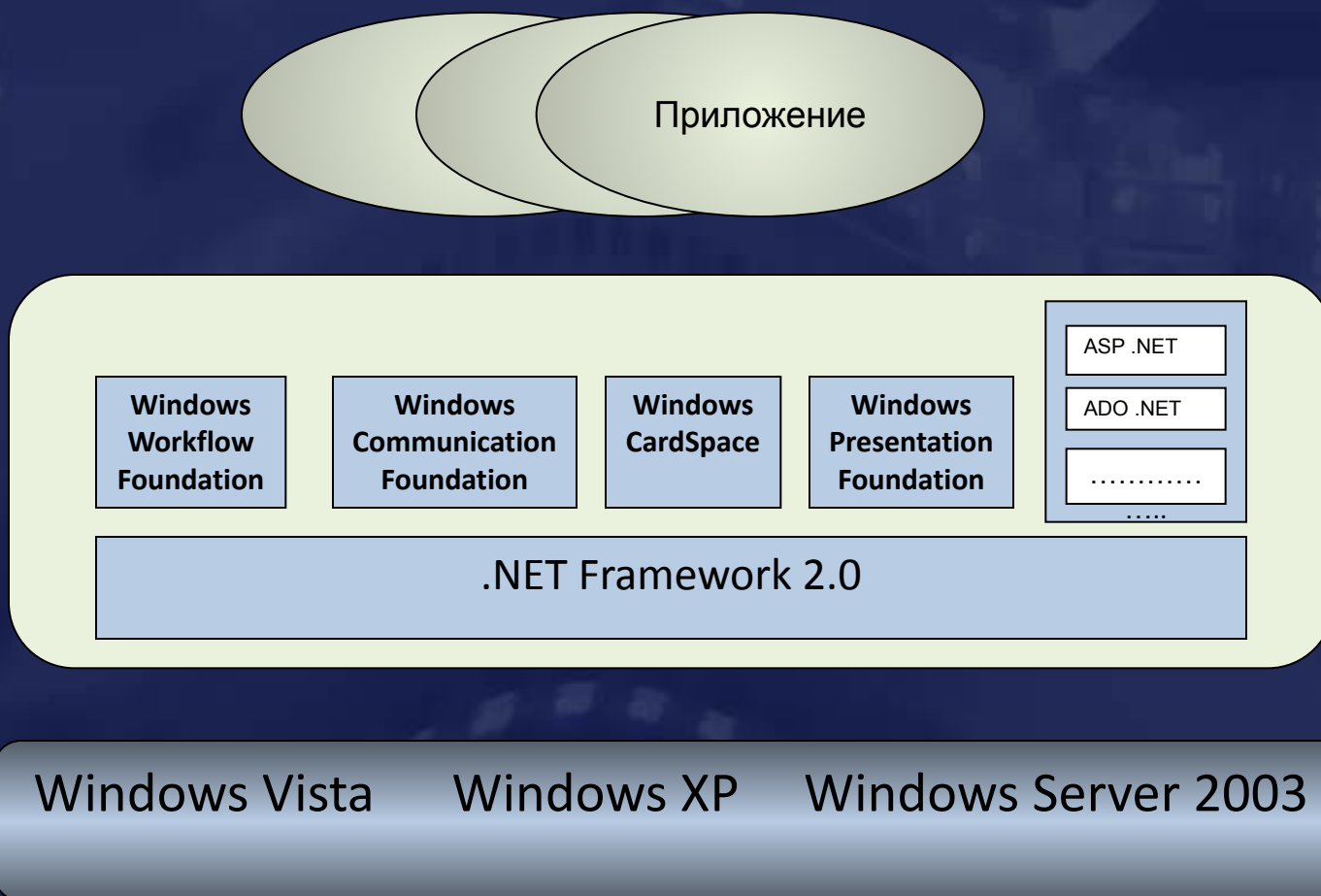


Литература

1. Троелсен Э. Язык программирования C# 2005 и платформа .NET 2.0 - Изд. Вильямс, 2007.
2. Мак-Дональд М. WPF: Windows Presentation Foundation в .NET 3.0 для профессионалов. - Изд. Вильямс, 2008.
3. Петцольд Ч. Microsoft Windows Presentation Foundation. Базовый курс. - Изд. Microsoft Press. Русская редакция, СПб.: Питер, 2008.
4. Нейгел К., Ивьен Б и др. C# 2005 и платформа .NET 3.0 для профессионалов. – Изд. Диалектика, 2007.

Платформа Microsoft® .NET 3.0 - 3.5

- ✓ Платформа Microsoft .NET Framework 3.0 — это новая модель программирования управляемого кода для операционной системы Windows.



Новые технологии платформы .NET 3.0 - 3.5

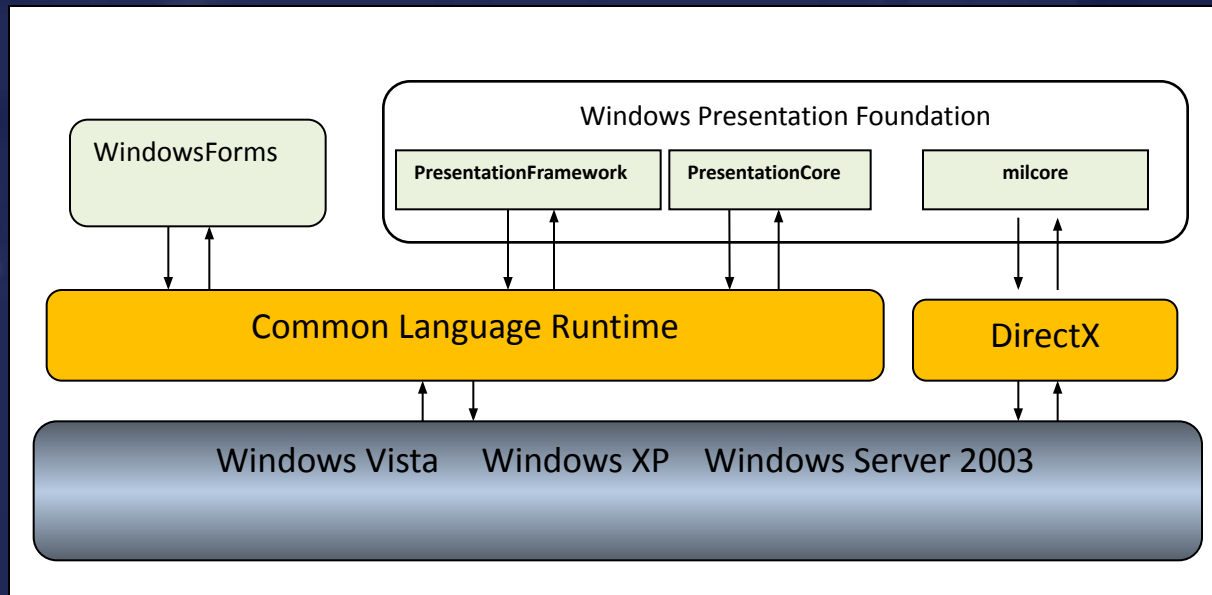
WWF - Windows Workflow Foundation – программная модель и набор инструментов для определения и выполнения логики потока работ независимо от специфики задействованных в нем операций.

WCF - Windows Communication Foundation – новая коммуникационная технология - управление процессом создания распределенных систем и взаимодействия компонентов по сети на базе протоколов Web-сервисов.

WPF – Windows Presentation Foundation – новая подсистема визуализации и пользовательского интерфейса.

Windows CardSpace – технология безопасной идентификации пользователей при перемещении между ресурсами Интернета без необходимости повторного ввода имен и паролей (развитие Microsoft Passport).

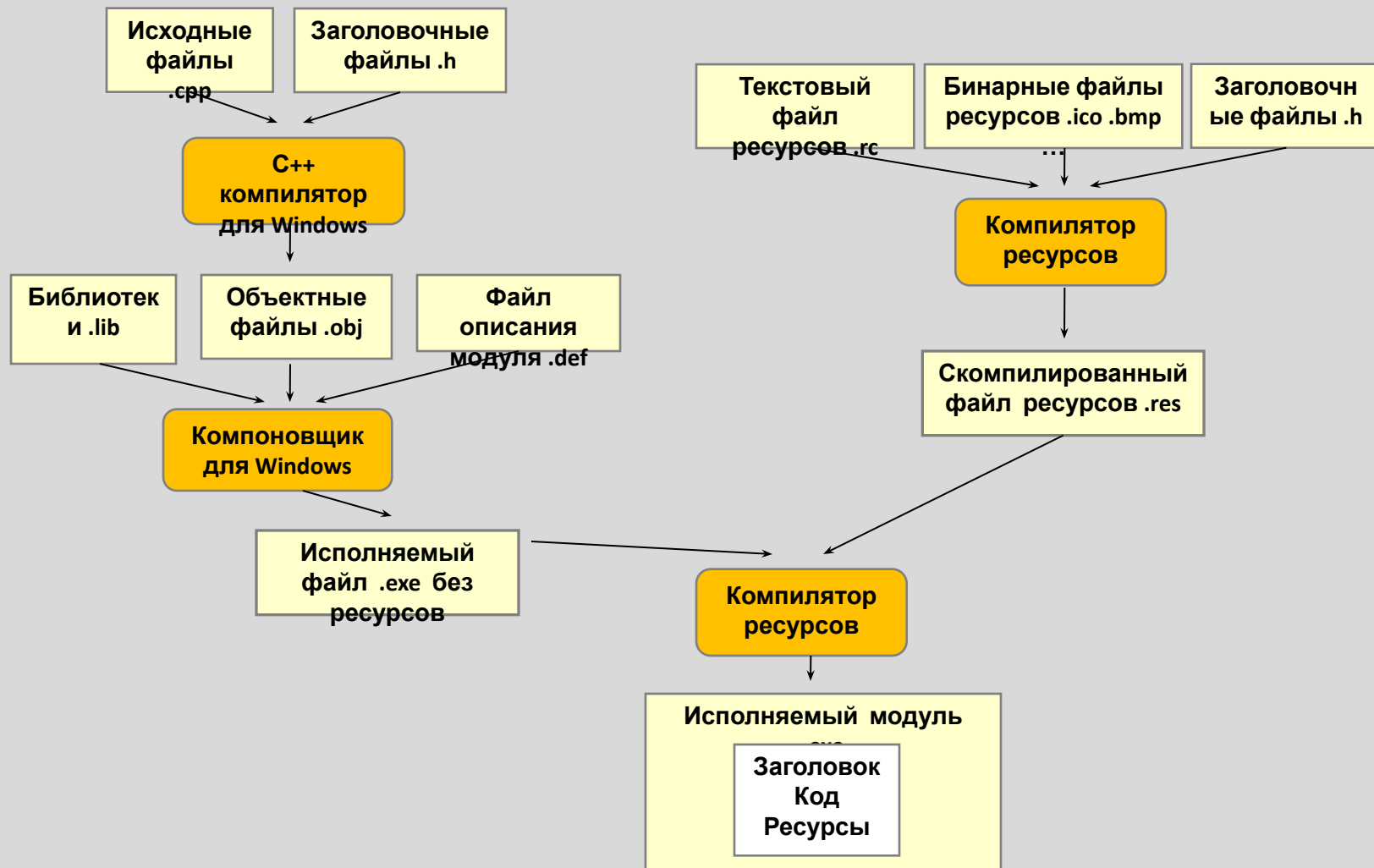
WPF- Windows Presentation Foundation



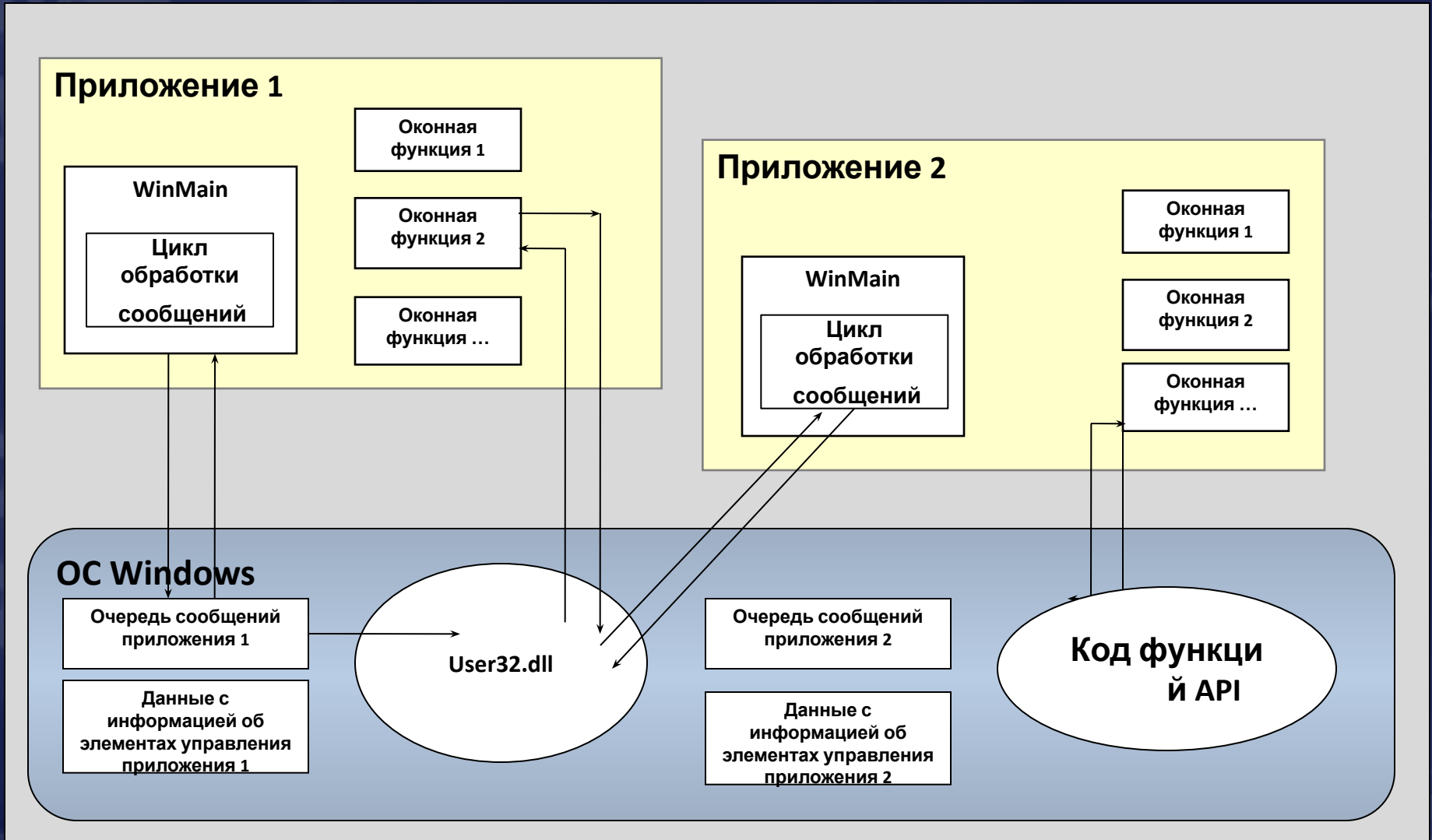
WPF – новая программная модель пользовательского интерфейса.

- Весь вывод происходит через DirectX – интерфейс аппаратно-ускоренной графики.
- Возможность использования языка разметки XAML (eXtensible Application Markup Language) для декларативного описания пользовательского интерфейса приложения.

Компиляция приложения Windows API



Выполнение приложения Windows API



Windows Forms

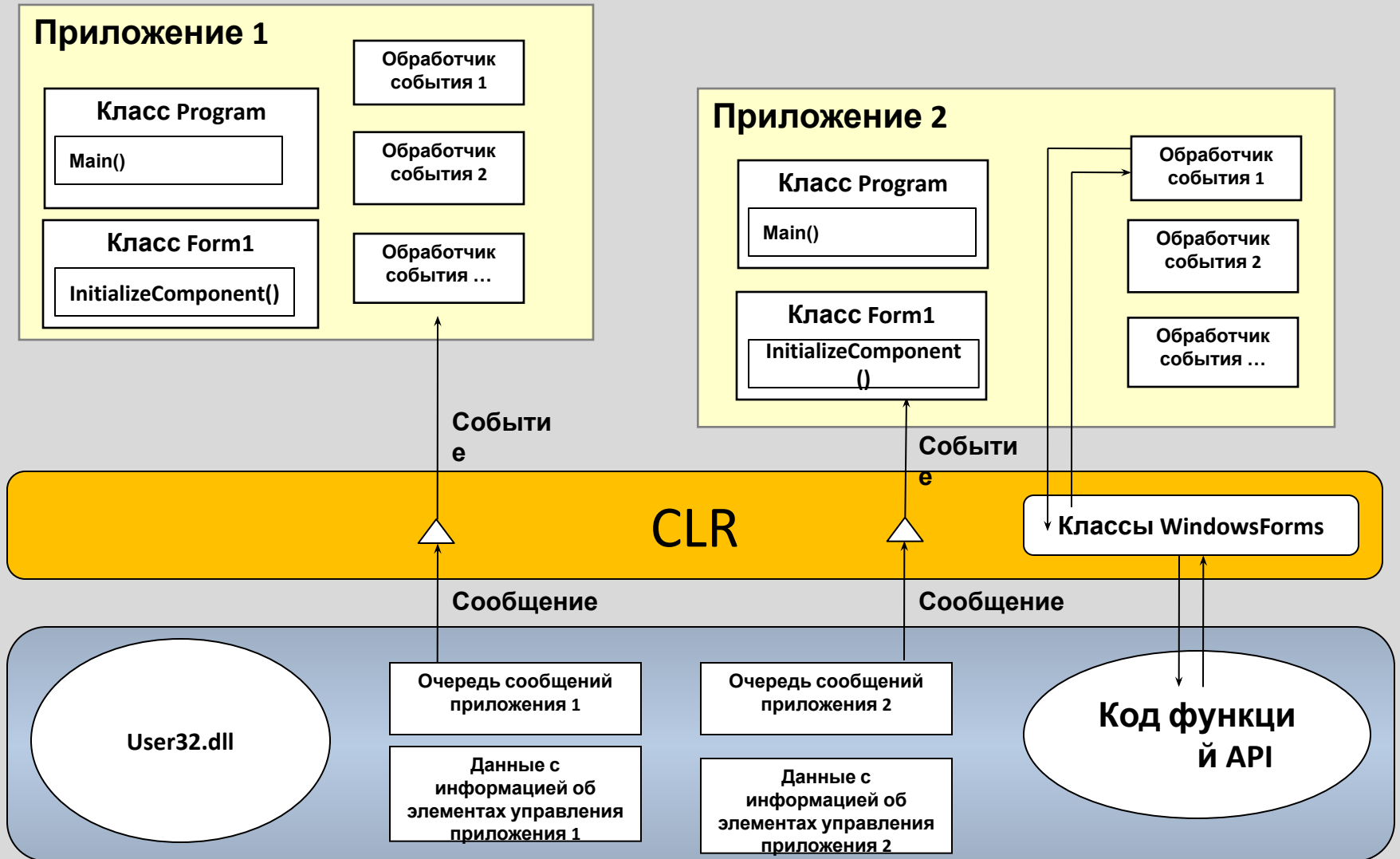
- ✓ WindowsForms - подмножество типов библиотеки BCL в .NET Framework 1.0, 2.0, 3.0 для создания пользовательского интерфейса локальных приложений.
- ✓ Классы поддерживают все элементы пользовательского интерфейса Win32.
- ✓ Некоторые классы описывают элементы управления, работающие только в .NET Framework.
- ✓ Большая часть типов определена в пространстве имен System.Windows.Forms и во вложенных в него пространствах имен .

Windows Forms и Win32

- ✓ Форма (Form) - окно верхнего уровня или окно диалога. Обычно окно верхнего уровня содержит несколько элементов управления - дочерних окон, реализующих типовую функциональность.
- ✓ Любое окно приложения является объектом операционной системы – при его создании ОС в своих сегментах распределяет память для информации о текущем состоянии окна и назначает ему дескриптор (handle).
- ✓ По дескрипторам операционная система идентифицирует окна приложений.
- ✓ В приложении на C# форма является объектом типа `System.Windows.Forms.Form` или производного от него.
- ✓ Дескриптор окна доступен через свойство базового класса `System.Windows.Forms.Control` и иногда используется программой.

```
public virtual IntPtr Handle {get;}
```


Выполнение приложения WindowsForms



Иерархия классов Windows Forms

`System.Object`

`System.MarshalByRefObject`

`System.ComponentModel.Component`

`System.Windows.Forms.Control`

`System.Windows.Forms.ScrollableControl`

`System.Windows.Forms.ContainerControl`

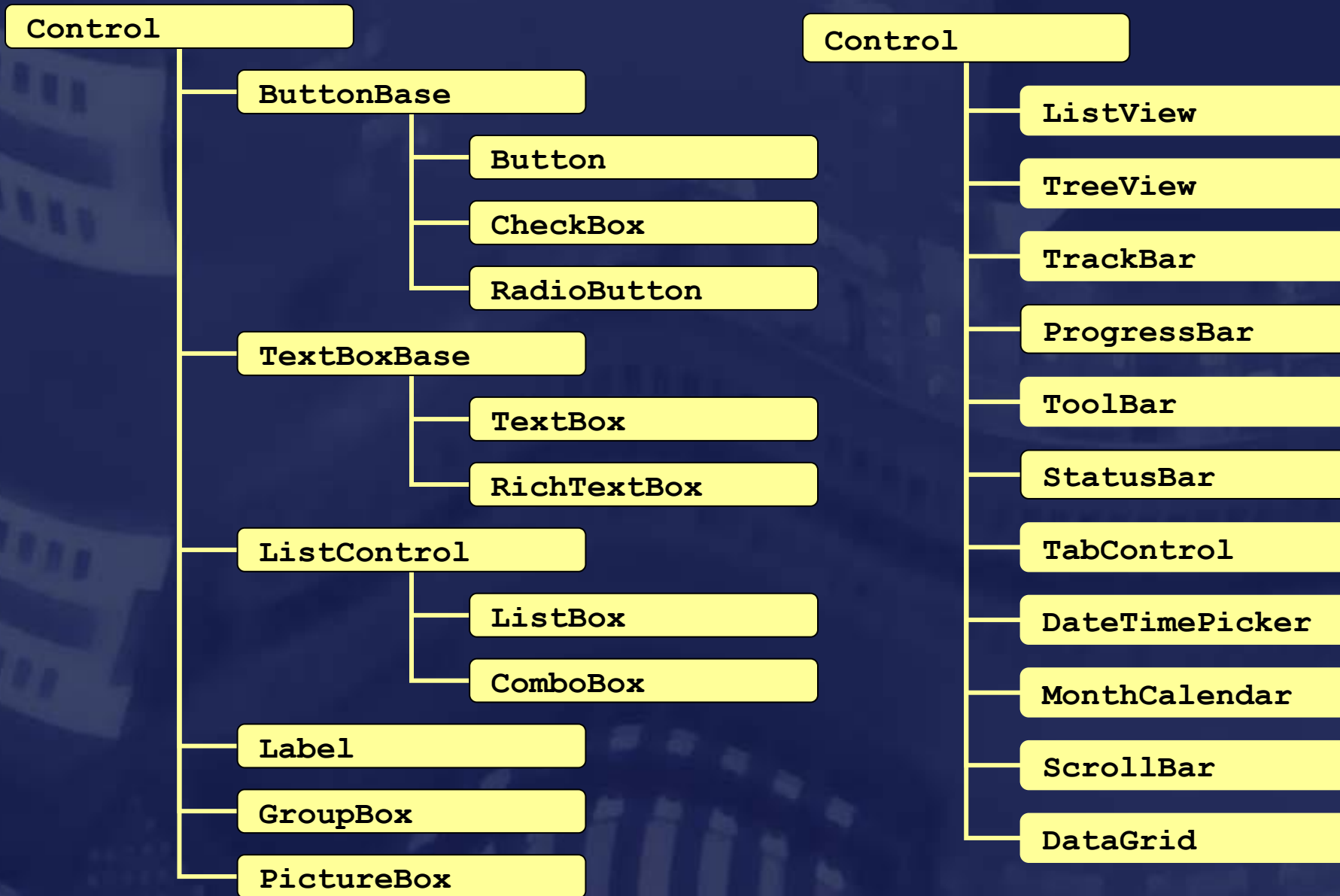
`System.Windows.Forms.Form`

<code>MarshalByRefObject</code>	Объекты передаются по ссылке через границу домена приложения, процесса или компьютера (в отличие от типов, производных от <code>MarshalByValueComponent</code>).
<code>Component</code>	реализует <code>IComponent</code>

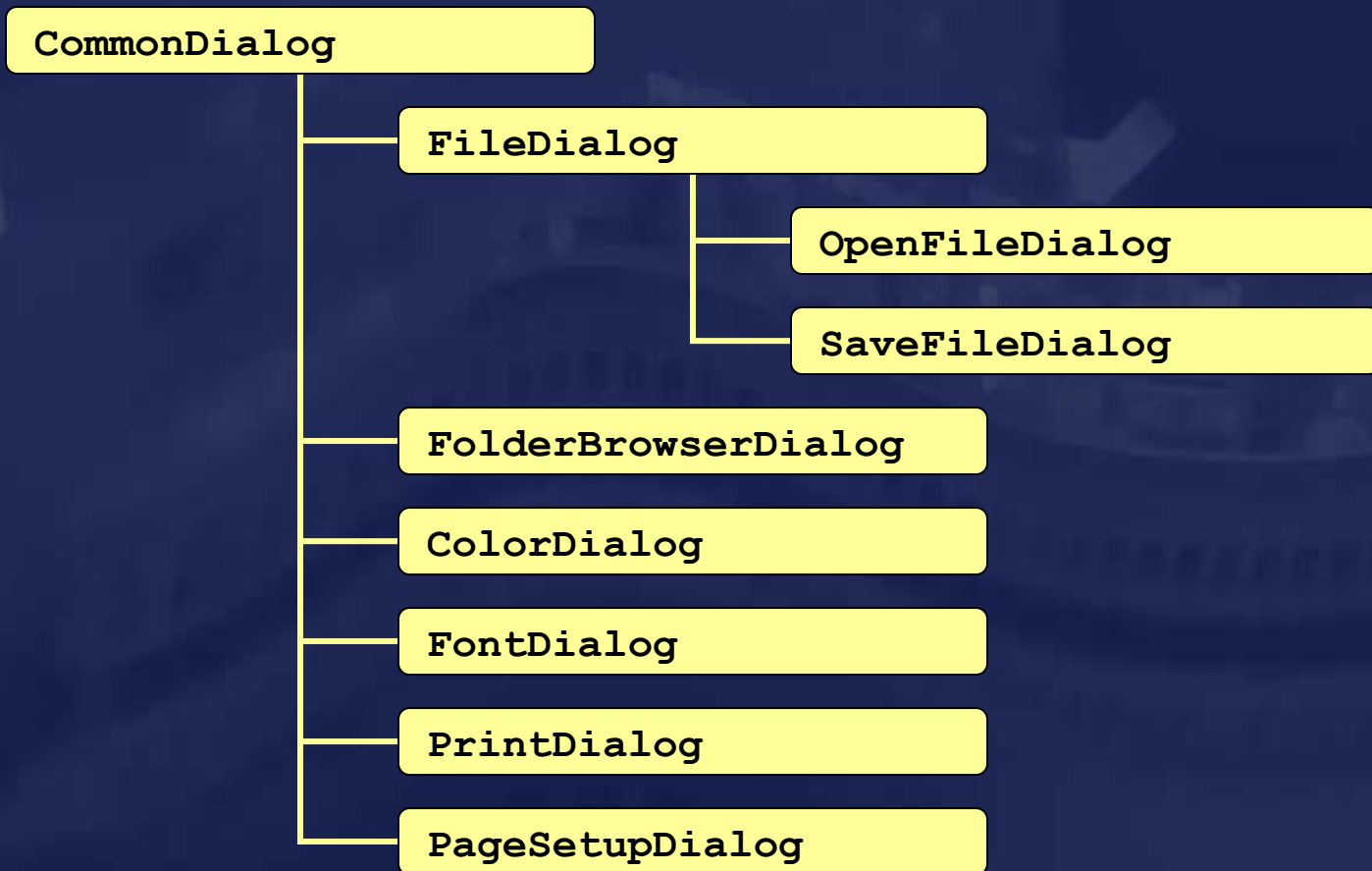
Иерархия классов Windows Forms -2

Control	Базовый для класса Form и классов элементов управления, реализует общую функциональность всех элементов управления. Компонент с визуальным представлением.
ScrollableControl	Базовый класс для элементов управления, поддерживающих автоматическую прокрутку (auto-scrolling). Содержит свойство <code>public virtual bool AutoScroll {get; set;}</code>
ContainerControl	Реализует IContainer. Логически содержит одну или несколько компонент, которые называются дочерними компонентами контейнера.

Классы элементов управления Windows Forms

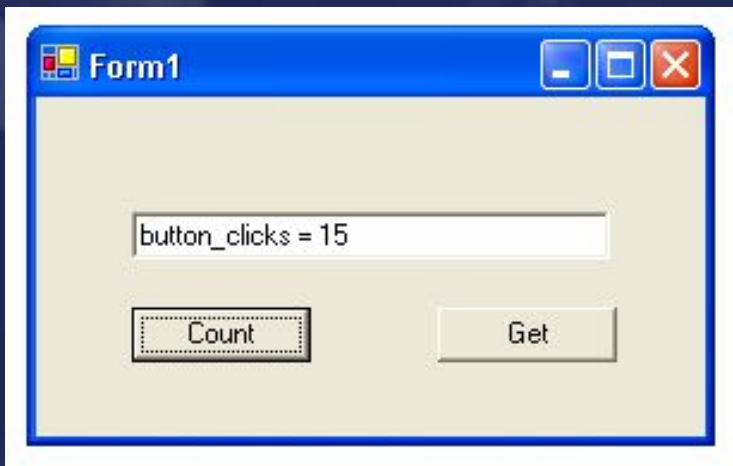


Классы стандартных диалогов



Визуальный дизайнер форм

При создании проекта Visual C# Windows Forms Application



- ✓ создает класс, производный от класса `System.Windows.Forms.Form` - `Form1` в примере;
- ✓ добавляет в класс поля - ссылки на объекты классов, описывающих элементы управления на форме.

```
public class Form1 : System.Windows.Forms.Form
{
    private System.Windows.Forms.Button button1;
    private System.Windows.Forms.Button button2;
    private System.Windows.Forms.TextBox textBox1;
    ...
}
```

Метод InitializeComponent()

- ✓ Визуальный дизайнер форм создает метод InitializeComponent() и размещает его вызов в конструкторе формы.

```
public Form1()  
{  
    //  
    // Required for Windows Form Designer support  
    //  
    InitializeComponent();  
}
```

- ✓ В методе InitializeComponent()
 - распределяется память под объекты классов для элементов пользовательского интерфейса и присваиваются значения по умолчанию некоторым их свойствам;
 - все элементы управления (ссылки на объекты) добавляются к коллекции элементов управления.

Метод InitializeComponent() -2

- ✓ В методе InitializeComponent() все элементы управления (ссылки на объекты) добавляются к коллекции элементов управления.

```
private void InitializeComponent()  
{ ...  
this.Controls.Add(this.button2);  
this.Controls.Add(this.textBox1);  
this.Controls.Add(this.button1);  
this.Name = "Form1";  
this.Text = "Form1";  
this.ResumeLayout(false);  
}
```

- ✓ Форма как окно(объект ОС) создается
 - либо при вызове статического метода Run класса Application,
 - либо при вызове экземплярных методов Show или ShowDialog класса Form

```
static void Main()  
{ Application.Run(new Form1()); }
```


События и свойства

- ✓ Свойства формы/элемента управления определяют внешний вид и режимы работы.
- ✓ События несут информацию обо всех изменениях, которые происходят с формой или элементом управления.
- ✓ Все свойства и события, определенные в классе элемента управления, отображаются в визуальном дизайнера форм.

Имя
класса
формы



Переключатели для
показа свойств или
событий,
отсортированных по
алфавиту или по
категориям.

Взаимодействие с пользователем

- ✓ Все аппаратные прерывания мыши и клавиатуры (действия пользователя) кодируются ОС в определенном формате и размещаются в очереди сообщений.
- ✓ В очереди сообщений размещаются также сообщения, связанные с работой самого приложения или других приложений.
- ✓ ОС для каждого приложения создает и поддерживает свою очередь сообщений.
- ✓ Каждое сообщение связано с конкретным окном приложения (хранит дескриптор окна) и может содержать дополнительную информацию.

Взаимодействие с .NET Framework

- ✓ .NET Framework выбирает сообщения из очереди приложения и преобразует их в события.
- ✓ Объекты приложения подписываются на события, указывая метод-обработчик события.
- ✓ Некоторые классы имеют обработчики по умолчанию.
- ✓ В приведенном ниже примере форма подписалась на три события – Load, FormClosing и FormClosed.

```
private void InitializeComponent()  
{ ...  
    this.Load += new System.EventHandler(this.Form1_Load) ;  
    this.FormClosed += new  
System.Windows.Forms.FormClosedEventHandler(this.Form1_FormClosed) ;  
    this.FormClosing += new  
System.Windows.Forms.FormClosingEventHandler(this.Form1_FormClosing) ;  
}
```

Обработчики событий

✓ Обработчику события .Net Framework передает два параметра.

- первый содержит ссылку на объект, который вызвал событие.
- через второй параметр передается ссылка на объект с дополнительной информацией о событии. Тип второго параметра зависит от события. Некоторые события (например, событие Button.Click) не передают дополнительную информацию.

```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show("button1_Click");
}

private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    if (first) {
        first = false;
        e.Cancel = true;
    }
    MessageBox.Show("Form1_FormClosing");
}

private void Form1_FormClosed(object sender, FormClosedEventArgs e)
{
    MessageBox.Show("Closed: " + e.CloseReason.ToString());
}
```

Окно сообщений. Класс MessageBox

- ✓ Для вывода сообщений и организации простейшего взаимодействия с пользователем обычно используется статический метод Show класса MessageBox

```
System.Object
```

```
System.Windows.Forms.MessageBox
```

- ✓ В классе только один метод Show (статический, 21 перегрузка в 3.5)

```
public static DialogResult Show( string text );  
  
public static DialogResult Show( string text, string caption,  
                                MessageBoxButtons buttons );  
  
public static DialogResult Show ( string text, string  
caption,  
                                MessageBoxButtons buttons,  
                                MessageBoxIcon icon,  
                                MessageBoxDefaultButton defaultButton);
```

Перечисления для MessageBox

- ✓ Перечисление `MessageBoxButtons` определяет набор кнопок в диалоге сообщений `MessageBox`.

<code>Ok</code>	<code>OkCancel</code>	<code>RetryCancel</code>
<code>YesNo</code>	<code>YesNoCancel</code>	
<code>AbortRetryIgnore</code>		

- ✓ Перечисление `MessageBoxIcons` определяет одну из готовых иконок в диалоге сообщений `MessageBox`.

<code>Asterisk</code>	<code>Error</code>	<code>Exclamation</code>
<code>Hand</code>	<code>Information</code>	<code>None</code>
<code>Question</code>	<code>Stop</code>	<code>Warning</code>

- ✓ Перечисление `MessageBoxDefaultButton` определяет кнопку по умолчанию в диалоге сообщений `MessageBox`.

<code>Button1</code>	<code>Button2</code>	<code>Button3</code>
----------------------	----------------------	----------------------

Перечисление DialogResult

используется

- как возвращаемое значение статического метода Show класса MessageBox;
- как значение свойства DialogResult класса Form;
- как значение свойства DialogResult класса Button.

Элементы перечисления DialogResult:

Ok	Cancel	
Yes	No	None
Abort	Retry	Ignore

```
if ( MessageBox.Show("Continue?","",  
    MessageBoxButtons.YesNo, MessageBoxIcon.Question,  
    MessageBoxDefaultButton.Button1) == DialogResult.Yes)  
    { ...  
    }  
else return;
```