

Диалоги

- ✓ В классах Windows Forms нет специального класса для диалогов. Диалоги представляют собой объекты класса `System.Windows.Forms.Form` и создаются с помощью методов :

```
public void Show();  
public DialogResult ShowDialog();  
public DialogResult ShowDialog( IWin32Window owner );
```

ShowDialog	<p>Создает модальный диалог. После вызова <code>ShowDialog</code> следующий оператор не выполняется пока не произойдет выход из метода.</p> <p>Пока не закрыт модальный диалог, форма-владелец не принимает ввод с клавиатуры и сообщения мыши, но принимает сообщения таймера и вызовы <code>OnPaint</code>.</p>
Show	<p>Создает немодальный диалог. Сразу же после создания окна диалога происходит выход из метода. Диалог <code>< Find and Replace ></code> в среде <code>VisualStudio</code> работает как немодальный диалог.</p>

Модальные диалоги

- ✓ Создаются при вызове метода ShowDialog класса System.Windows.Forms.Form.
- ✓ Закрываются, когда свойству DialogResult формы, которая описывает диалог,

```
public DialogResult DialogResult {get; set;}
```

присваивается одно из значений перечисления DialogResult.

- ✓ Если закрыть диалог, вызвав метод Close(), то свойство будет иметь значение DialogResult.Cancel.
- ✓ Модальный диалог обычно закрывается другим способом – когда свойству DialogResult кнопки на форме присваивается одно из значений перечисления DialogResult. Это же самое значение получает свойство DialogResult формы.

Модальные диалоги -2

В примере модальный диалог (Form3) закрывается при нажатии кнопки <Yes>, т.к. свойству DialogResult кнопки при инициализации присвоено значение перечисления DialogResult, и при нажатии кнопки <No>, так как в обработчике для кнопки <No> свойству DialogResult формы присваивается значение перечисления DialogResult.

```
public class Form3 : System.Windows.Forms.Form
{
    private System.Windows.Forms.Button button1;
    private System.Windows.Forms.Button button2;

    private void InitializeComponent()
    {
        [...code...]
        // button1
        this.button1.DialogResult = System.Windows.Forms.DialogResult.Yes;
        this.button1.Name = "button1";
        this.button1.Text = "Yes";
        [...code...]
        // button2
        this.button2.Name = "button2";
        this.button2.Text = "No";
        this.button2.Click += new System.EventHandler(this.button2_Click);
        [...code...]
    }

    private void button2_Click(object sender, System.EventArgs e)
    {this.DialogResult = DialogResult.No; }
```

Немодальные диалоги

- ✓ Немодальный диалог создается при вызове метода `Show()` класса `System.Windows.Forms.Form`.
- ✓ Закрывается вызовом метода `Close()`; Метод `Close()` для формы освобождает ресурсы (`disposed`). Если затем форма создается снова, необходимо опять распределить память методом `new`.
- ✓ В отличие от метода `Close()` метод `Hide()` не освобождает ресурсы.
- ✓ Объект класса `System.Windows.Forms.Form`, который описывает диалог, должен быть полем или свойством формы-владельца.

Немодальные диалоги. Пример

✓ Пример ModelessDialog. Класс Form1 для основной формы

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

    Form2 dlg ;
    public string TextInForm1
    {
        get { return textBox1.Text; }
        set { textBox1.Text = value; }
    }

    public bool EnableDialogButton
    {
        get { return button_dialog.Enabled; }
        set { button_dialog.Enabled = value; }
    }
    [... code...]
}
```

Немодальные диалоги. Пример -2

- ✓ Обработчик события в Form1, в котором создается немодальный диалог:

```
public class Form1 : System.Windows.Forms.Form
{
    [...code...]

    private void button_dialog_Click(object sender,
                                     System.EventArgs e)
    {
        button_dialog.Enabled = false;
        dlg = new Form2();
        dlg.Owner = this;
        dlg.StartPosition = FormStartPosition.Manual;
        dlg.Location = new Point( this.Right-this.Right/5,
                                this.Bottom-this.Bottom/5);

        dlg.Show();
        MessageBox.Show("After dlg.Show();");
    }
}
```

Немодальные диалоги. Пример -3

✓ Пример ModelessDialog. Класс Form2 для диалога:

```
public class Form2 : System.Windows.Forms.Form
{
    [... code...]
    private System.Windows.Forms.Button button_apply;
    private System.Windows.Forms.TextBox textBox1;

    private void button_apply_Click( object sender,
                                     System.EventArgs e)
    { ((Form1)this.Owner).TextInForm1 = this.textBox1.Text;
    }

    private void Form2_Closed(object sender, System.EventArgs e)
    { ((Form1)this.Owner).EnableDialogButton = true;
    }
}
```

Стандартные диалоги (Common dialog boxes)

`System.Object`

`System.MarshalByRefObject`

`System.ComponentModel.Component`

`System.Windows.Forms.CommonDialog`

`System.Windows.Forms.ColorDialog`

`System.Windows.Forms.FileDialog`

`System.Windows.Forms.OpenFileDialog`

`System.Windows.Forms.SaveFileDialog`

`System.Windows.Forms.FolderBrowserDialog`

`System.Windows.Forms.FontDialog`

`System.Windows.Forms.PageSetupDialog`

`System.Windows.Forms.PrintDialog`

Классы OpenFileDialog и SaveFileDialog

- ✓ Классы используются для создания стандартных диалогов для выбора или ввода имени файла. Диалоги дают возможность навигации по файловой системе.
- ✓ Диалоги создаются только как модальные. После того, как диалог закрыт, имя выбранного пользователем файла доступно через свойства объекта класса OpenFileDialog (SaveFileDialog).
- ✓ Определенные в классах OpenFileDialog и SaveFileDialog свойства дают возможность немного изменить набор элементов и функциональность стандартных диалогов.

Некоторые открытые свойства класса OpenFileDialog

<code>string DefaultExt {get; set;}</code>	Умолчание для расширения имени файла.
<code>bool AddExtension {get; set;}</code>	Автоматическое добавление расширения имени файла.
<code>virtual bool CheckFileExists {get; set;}</code>	Вывод предупреждения, если пользователь ввел имя несуществующего файла.
<code>bool CheckPathExists {get; set;}</code>	Вывод предупреждения, если не существует путь.
<code>string FileName {get; set;}</code>	Полное имя файла, выбранное или введенное пользователем .
<code>string[] FileNames {get;}</code>	Массив имен файлов, выбранных пользователем.
<code>string Filter {get; set;}</code>	Строка фильтра.
<code>int FilterIndex {get; set;}</code>	Индекс пары <описание-фильтр> из строки фильтра, которая будет выведена при создании диалога.

Некоторые открытые свойства класса OpenFileDialog -2

<code>string InitialDirectory {get; set;}</code>	Каталог, который будет выведен при создании диалога.
<code>bool RestoreDirectory {get; set;}</code>	Восстановление текущего каталога после того, как диалог будет закрыт.
<code>bool ShowHelp {get; set;}</code>	Наличие кнопки Help, при нажатии кнопки происходит событие HelpRequested.
<code>string Title {get; set;}</code>	Заголовок диалога.

✓ Можно восстановить все значения свойств, принятые по умолчанию, вызвав метод

```
public override void Reset();
```

Пример

✓ В приведенном ниже коде

- создается объект OpenFileDialog;
- задается строка-фильтр;
- в том случае, когда пользователь подтвердил выбор, нажав кнопку Open, имя файла выводится в элемент управления TextBox.

```
private void button_Open_Click( object sender,
                                System.EventArgs e)
{
    OpenFileDialog dlg = new OpenFileDialog();
    dlg.Filter =
        "C#(*cs)|*.cs|Images(*.bmp;*.gif)|*.bmp;*.gif;|All(*.*)|*.*";
    dlg.FilterIndex = 1;

    if ( dlg.ShowDialog()==DialogResult.OK )
    { textBox2.Text = dlg.FileName;}
}
```

Класс OpenFileDialog

✓ Свойства класса OpenFileDialog:

<code>bool Multiselect {get; set;}</code>	Разрешение на выбор нескольких имен файлов.
<code>bool ShowReadOnly {get; set;}</code>	Наличие отмечаемой кнопки ReadOnly.
<code>bool ReadOnlyChecked {get; set;}</code>	Состояние отмечаемой кнопки ReadOnly.

✓ Метод класса открывает файл, имя которого выбрал пользователь, в режиме только для чтения (независимо от состояния отмечаемой кнопки <ReadOnly>) :

```
public Stream OpenFile();
```

Класс SaveFileDialog

✓ Свойства класса SaveFileDialog:

<code>bool CreatePrompt {get; set;}</code>	Запрос, следует ли создавать файл, который не существует.
<code>bool OverwritePrompt {get; set;}</code>	Запрос, следует ли перезаписать файл, который уже существует.

✓ Метод класса создает и открывает файл, имя которого выбрал пользователь, в режиме для чтения/записи (если файл уже существует, записанные в нем данные будут потеряны):

```
public Stream OpenFile();
```

Класс TabControl

- ✓ Класс TabControl описывает элемент управления, состоящий из нескольких страниц (класс TabPage). Каждая страница содержит свой набор элементов управления.
- ✓ Страницы заменяют друг друга при переключении с помощью закладок. С TabControl можно связать список изображений (ImageList) и вывести изображения на закладках.

```
System.Object
  System.MarshalByRefObject
    System.ComponentModel.Component
      System.Windows.Forms.Control
        System.Windows.Forms.TabControl
```

- ✓ В классе TabControl определено свойство TabPages, которое дает доступ к коллекции объектов, описывающих отдельные страницы диалога:

```
TabControl.TabPageCollection TabPages {get;}
```

Класс TabPage

- ✓ Класс TabPage описывает отдельную страницу диалога TabControl:

```
System.Object
  System.MarshalByRefObject
    System.ComponentModel.Component
      System.Windows.Forms.Control
        System.Windows.Forms.ScrollableControl
          System.Windows.Forms.Panel
            System.Windows.Forms.TabPage
```

- ✓ Класс TabPage является производным от класса Control и наследует все его свойства и методы.
- ✓ Открытые свойства класса TabPage:

```
public int ImageIndex
{get; set;}
```

Индекс изображения, который выводится на закладку для страницы.

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

Текст на закладке страницы.

```
string ToolTipText
{get; set;}
```

Текст подсказки для страницы.

События класса TabControl

- ✓ При переключении между страницами происходит событие:

```
public event EventHandler SelectedIndexChanged;
```

- ✓ Номер выбранной страницы можно получить через свойство SelectedIndex класса TabControl:

```
private void tabControl1_SelectedIndexChanged(object sender,  
                                             System.EventArgs e)  
{ if ( tabControl1.SelectedIndex == 1) { ... }  
}
```

- ✓ При генерации кода визуальный дизайнер размещает в классе, который описывает форму, содержащую элемент управления TabControl, ссылки на элементы управления со всех страниц TabControl.