Department of Economic Informatics and Cybernetics
Bucharest University of Economic Studies

Windows Applications Programming

Windows Forms

Liviu-Adrian Cotfas, PhD.

liviu.cotfas@ase.ro

Few words about me…

https://ro.linkedin.com/in/cotfasliviu

Further Reading / Watching

- Courses on **Microsoft Virtual Academy** - mva.microsoft.com
  - Free

- Courses on **PluralSight** - www.pluralsight.com
  - Free trial
  - Free access (limited period) through Microsoft DreamSpark

API reference and Source code

- API reference:
  - https://msdn.microsoft.com/en-us/library/

- .NET Framework source code:
  - http://referencesource.microsoft.com/#mscorlib/system/string.cs,8281103e6f23cb5c

Windows Forms

.NET Graphical User Interface

- Windows Forms (2001)
- Windows Presentation Foundation
- Universal Windows Platform

Startup and Shutdown

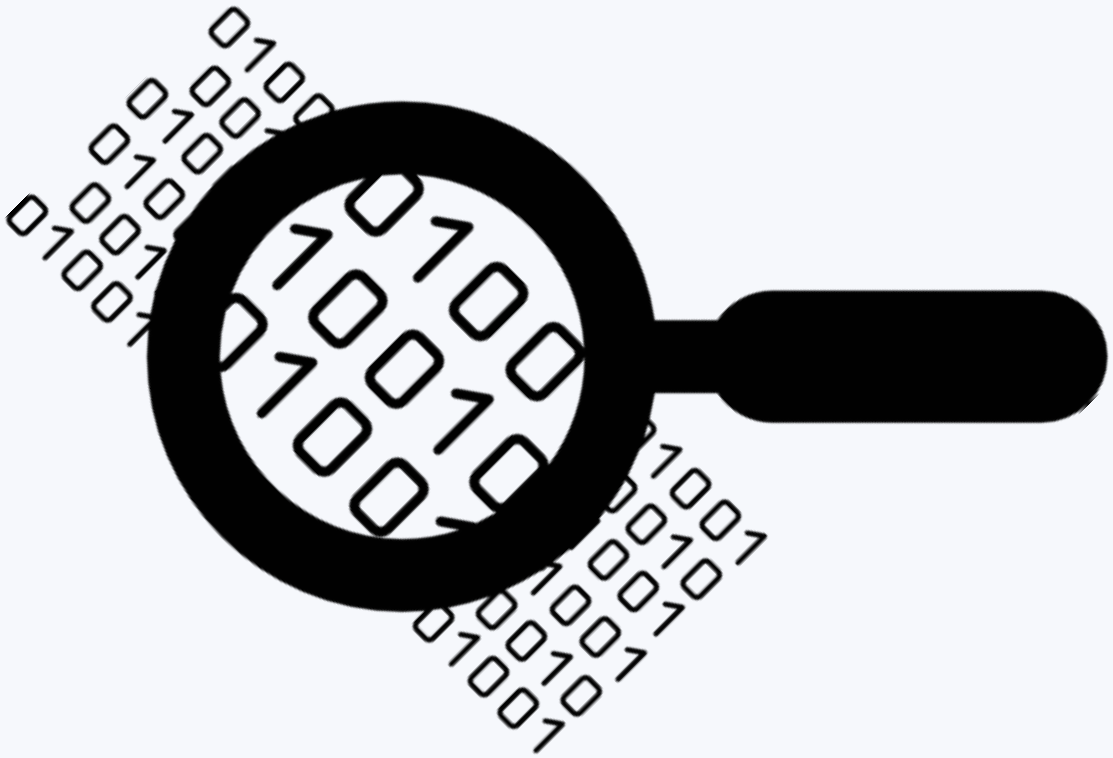
- Startup

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

Application Class

- The Application Class
  - Run
  - Exit
  - ThreadException

Demo



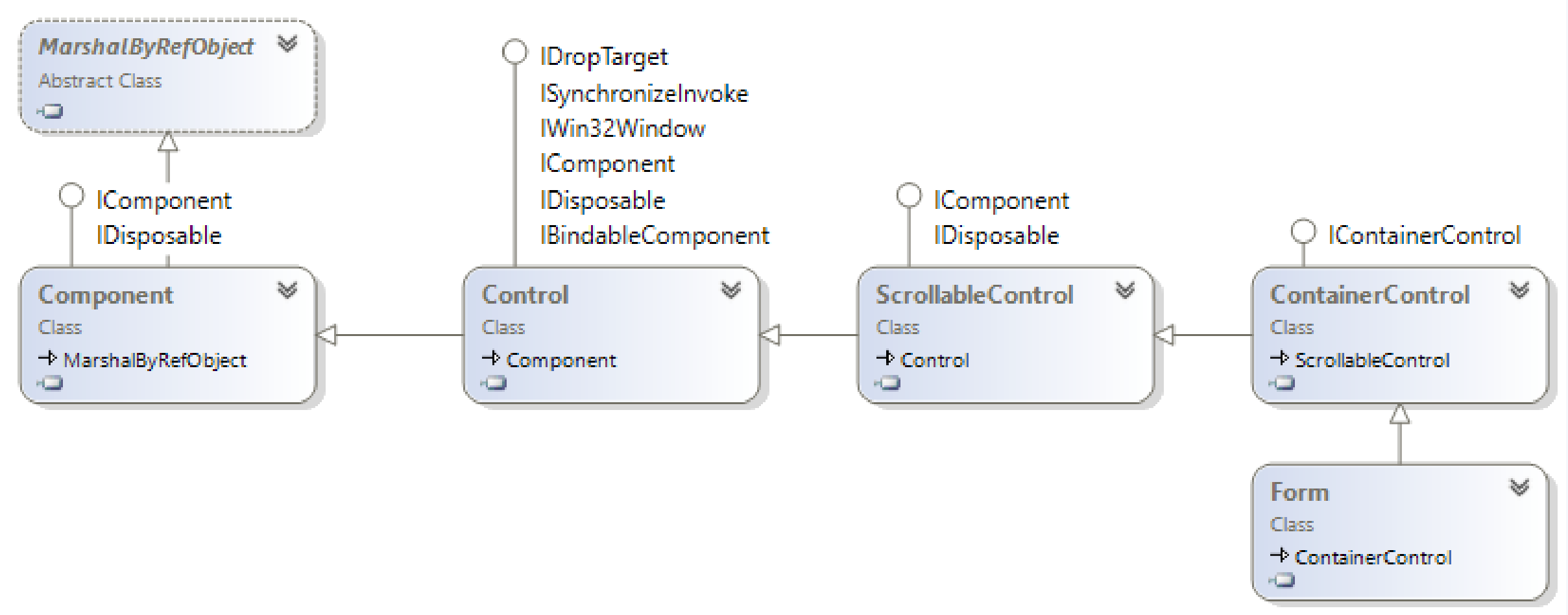
Form Class

Form Class

- All windows in a Windows Forms application are represented by objects of some type deriving from the Form class.

```
public partial class MainForm : Form
{
        // Constructor
        public MainForm()
        {
                InitializeComponent();
        }
}
```

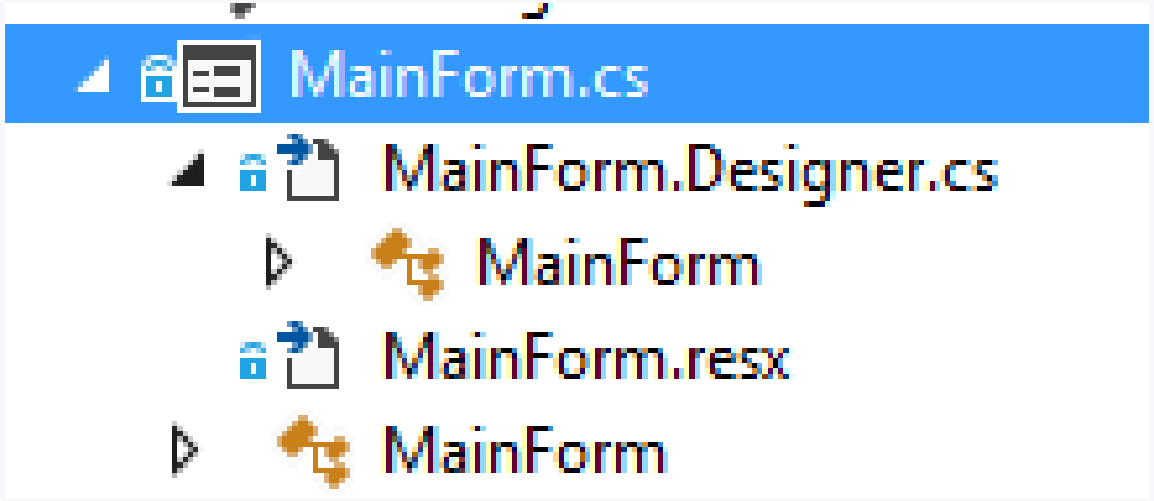
Inheritance

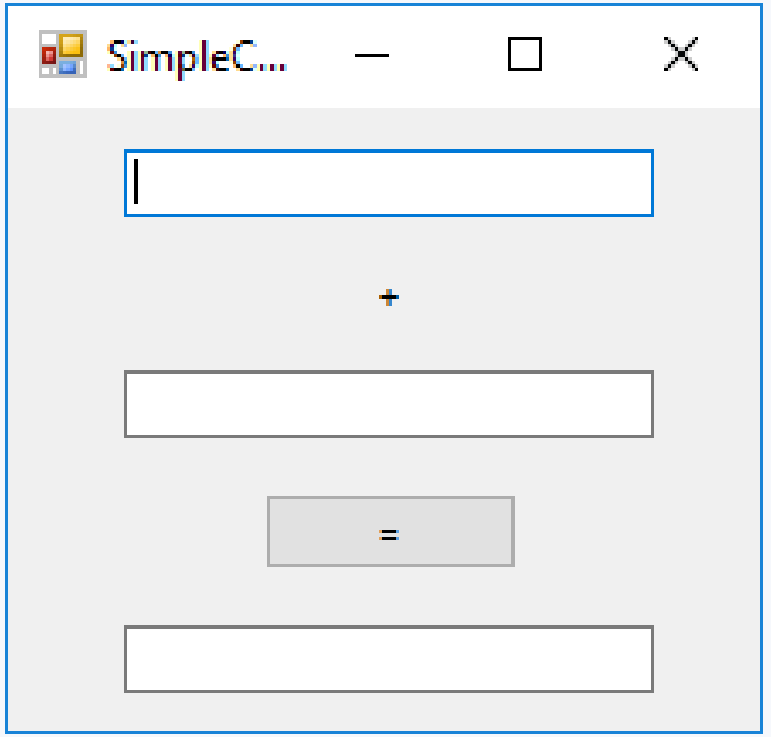


Further reading: link

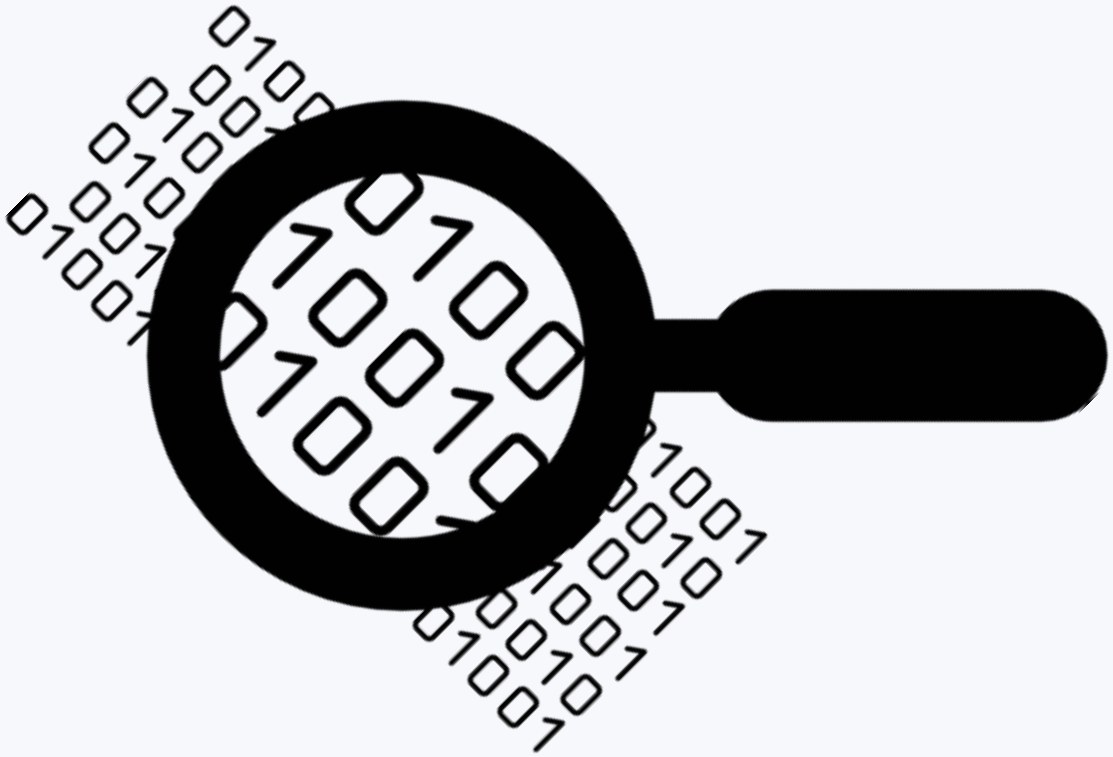
The Forms Designer

- visual designer that auto-generates code.
- Uses **partial classes**.





Demo



Partial Class

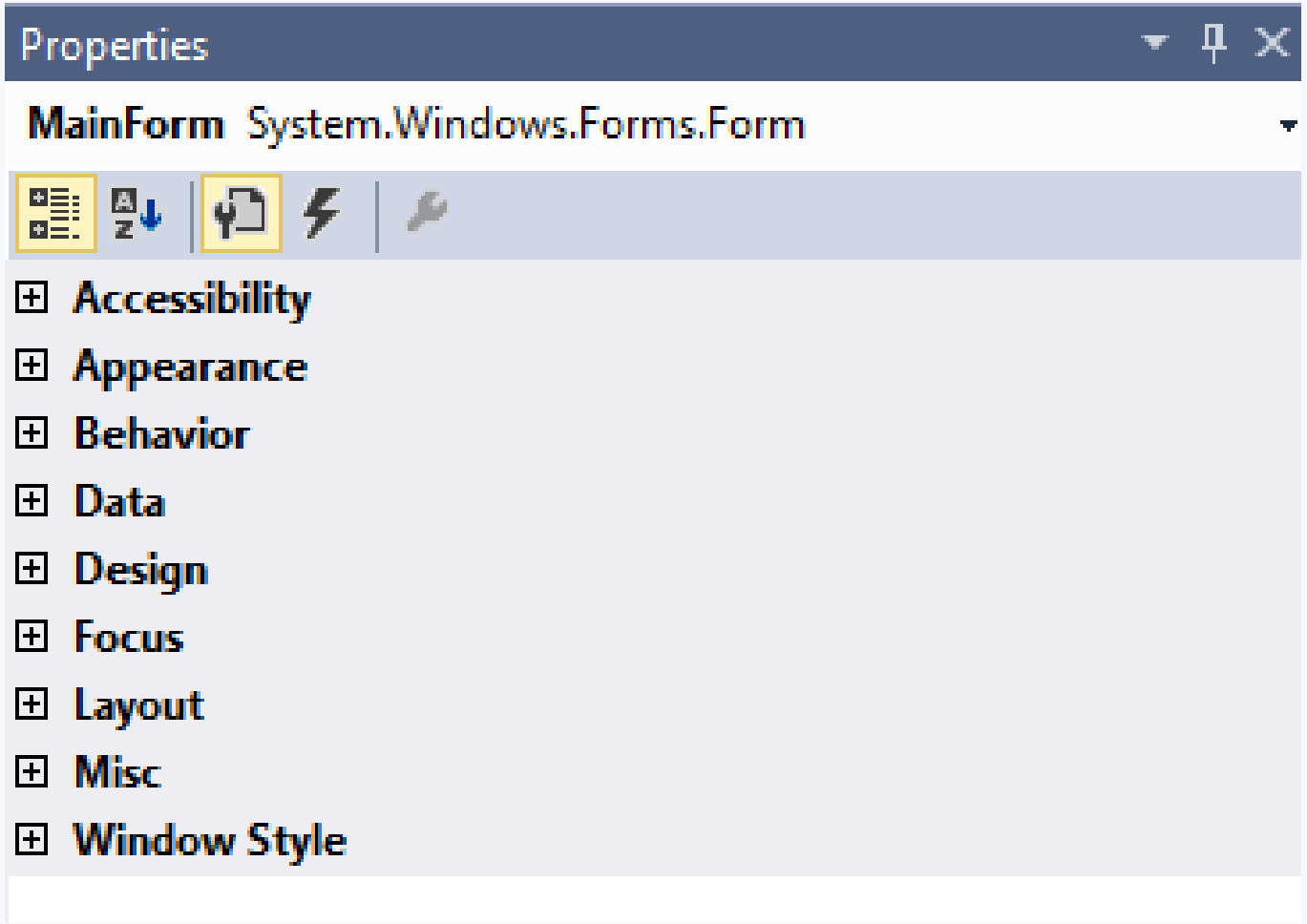
- It is possible to split the definition of a class or a struct, an interface or a method over two or more source files. Each source file contains a section of the type or method definition, and all parts are combined when the application is compiled.

```
public partial class Employee{
    public void DoWork()    {
    }
}

public partial class Employee{
    public void GoToLunch()    {
    }
}
```

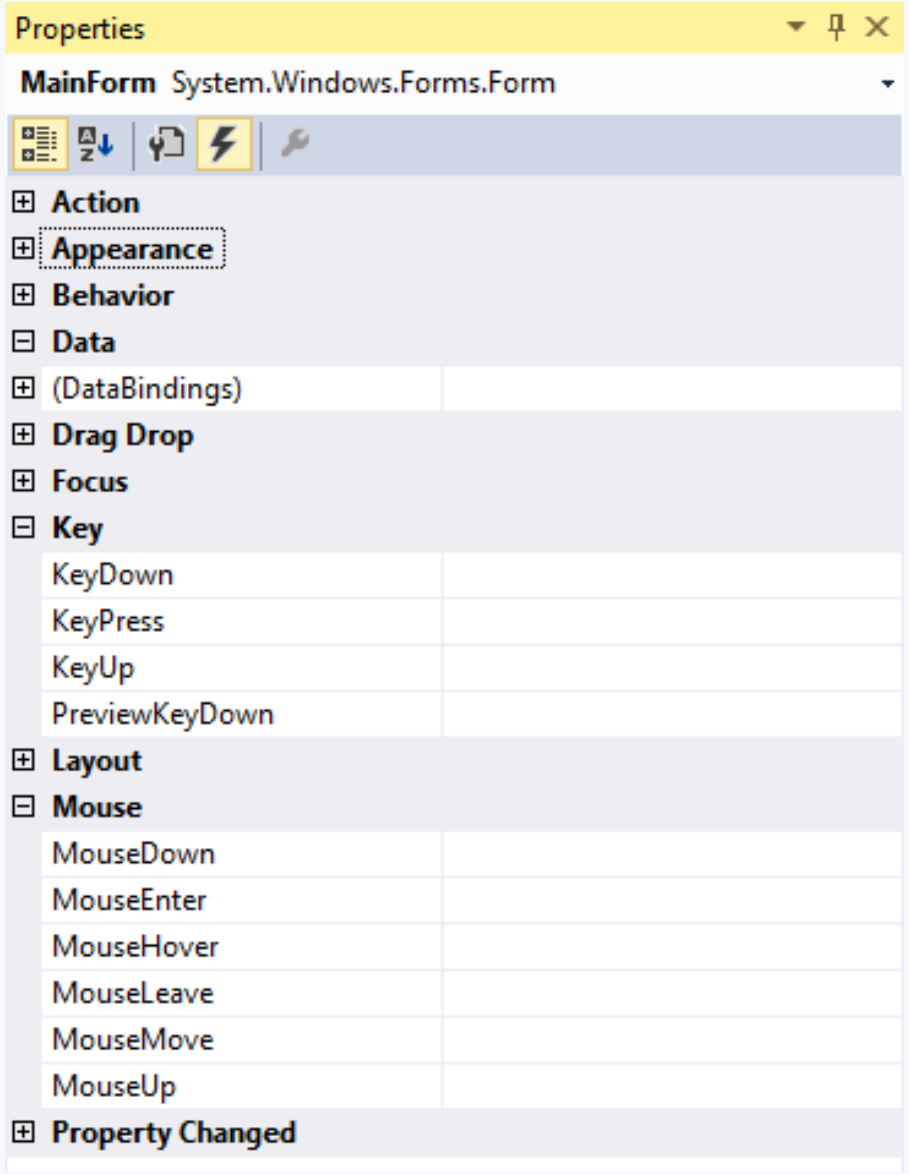
Properties

- Appearance
  - BackColor
  - Font
  - ForeColor
  - Text
- Design
  - Name



Events

- Key
  - KeyDown
  - KeyPress
  - KeyUp
- Mouse
  - MouseDown
  - MouseUp
  - Click



Keyboard Events

Keyboard Events

- Key events occur in the order specified bellow.

Event	Event data	Description
KeyDown	KeyEventArgs	Raised when a key is pressed. The KeyDown event occurs prior to the KeyPress event.
KeyPress	KeyPressEventArgs	Raised when a character generating key is pressed. The KeyPress event occurs after the KeyDown event and before the KeyUp event.
KeyUp	KeyEventArgs	Raised when a key is released.

KeyPress

- Occurs when a character is pressed on the keyboard, and again each time the character is repeated while it continues to be pressed.

- The KeyPress event is not raised by non-character keys other than space and backspace; however, the non-character keys do raise the KeyDown and KeyUp events.

- Use the KeyChar property to sample keystrokes at run time and to consume or modify a subset of common keystrokes.

KeyPressEventArgs

Property	Description
Handled	Boolean value indicating if the event was handled. false until set otherwise. When true, the keystroke is not displayed.
KeyChar	Read-only value of type char containing the composed ASCII character.

KeyDown and KeyUp

- **KeyDown** - Occurs when a key on the keyboard is pressed down.
- **KeyUp** - Occurs when a key on the keyboard is released.
- The **KeyDown** and **KeyUp** events are useful to fine-tune an application's behavior as keyboard keys are pressed and released, and for handling noncharacter keys such as the function or arrow keys.
- Handlers for these events receive an instance of the KeyEventArgs class as their event parameter.

KeyEventArgs properties

Property	Data type	Description
Alt	Boolean	Read-only value indicating if the Alt key was pressed. true if pressed, false otherwise.
Control	Boolean	Read-only value indicating if the Ctrl key was pressed. true if pressed, false otherwise.
Shift	Boolean	Read-only value indicating if the Shift key was pressed. true if pressed, false otherwise.
Modifiers	Keys	Read-only flags indicating the combination of modifier keys (Alt, Ctrl, Shift) pressed. Modifier keys can be combined using the bitwise OR operator.
Handled	Boolean	Value indicating if the event was handled. false until set otherwise.
KeyCode	Keys	Read-only value containing the key code for the key pressed. Typical values include the A key, Alt, and BACK (backspace).
KeyData	Keys	Read-only value containing the key code for the key pressed, combined with modifier flags to indicate combination of modifier keys (Alt, Ctrl, Shift).
KeyValue	integer	Key code property expressed as a read-only integer.

Mouse Events

Mouse Events

Event	Event argument	Description
Click	EventArgs	Raised when the control is clicked.
DoubleClick	EventArgs	Raised when the control is double-clicked.
MouseEnter	EventArgs	Raised when the mouse cursor enters the control.
MouseHover	EventArgs	Raised when the mouse cursor hovers over the control.
MouseLeave	EventArgs	Raised when the mouse cursor leaves the control.
MouseDown	MouseEventArgs	Raised when the mouse cursor is over the control and a mouse button is pressed.
MouseMove	MouseEventArgs	Raised when the mouse cursor is moved over the control.
MouseWheel	MouseEventArgs	Raised when the control has focus and the mouse wheel is rotated.
MouseUp	MouseEventArgs	Raised when the mouse cursor is over the control and a mouse button is released.

MouseEventArgs properties

Property	Description
Button	Returns the pressed mouse button. Must be one of the members of the MouseButtons enumeration
Clicks	Returns the integer number of times the mouse button was pressed and released. Resets after two clicks.
Delta	Returns the signed integer number of detents the mouse wheel was rotated. A positive value indicates that the wheel was rotated forward, i.e., away from the user, and a negative value indicates the wheel was rotated backward, i.e., toward the user.
X	The X coordinate, in pixels, of the mouse cursor's hot spot when the button was clicked, relative to the top-left corner of the control.
Y	The Y coordinate, in pixels, of the mouse cursor's hot spot when the button was clicked, relative to the top-left corner of the control.

Click

```
this.btnCalculate.Click += new System.EventHandler(this.btnCalculate_Click);
```

```
private void btnCalculate_Click(object sender, EventArgs e)
{
        var value1 = int.Parse(tbValue1.Text);
        var value2 = int.Parse(tbValue2.Text);

        var sum = value1 + value2;
        tbSum.Text = sum.ToString();
}
```

Alt Shortcuts

Alt Shortcuts

- An **access key** is an underlined character in the text of a menu, menu item, or the label of a control such as a button. It allows the user to "click" a button by pressing the ALT key in combination with the predefined access key.

```
// C#
// Set the letter "P" as an access key.
button1.Text = "&Print";
```

Secondary Forms

Modal & Modeless

- Forms can exhibit either **modal** or **modeless** behavior
- A **modal form** is one that demands the user's immediate attention, and blocks input to any other windows the application may have open.
- Good UI design suggests that you should use **modal** dialogs **only** when absolutely necessary. Typical examples: error messages

Modal & Modeless

```
//modal
Form frm = new Form(  );
frm.ShowDialog(  );

//modeless
Form frm = new Form(  );
frm.Show (  );
```

Resizing Forms

Menus

Exception Handling

Common Exception Types

- System.ArgumentException - Thrown when a function is called with a bogus argument.

- System.ArgumentNullException - Subclass of ArgumentException that's thrown when a function argument is (unexpectedly) null.

- System.ArgumentOutOfRangeException - Subclass of ArgumentException that's thrown when a (usually numeric) argument is too big or too small. For example, this is thrown when passing a negative number into a function that accepts only positive values.

Common Exception Types

- System.InvalidOperationException - Thrown when the state of an object is unsuitable for a method to successfully execute, regardless of any particular argument values. Examples include reading an unopened file or getting the next element from an enumerator where the underlying list has been modified partway through the iteration.

- System.NotSupportedException - Thrown to indicate that a particular functionality is not supported. A good example is calling the Add method on a collection for which IsReadOnly returns true.

- System.NotImplementedException - Thrown to indicate that a function has not yet been implemented.