Windows Forms and Basic Controls

Canberra Institute of Technology

aie

SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

Contents

- Windows Forms Overview
- Creating a New Windows Form
- Event Driven Programming
- Creating Event Handlers in Windows Forms
- Windows Forms Controls
- User Input
- Dialog Boxes

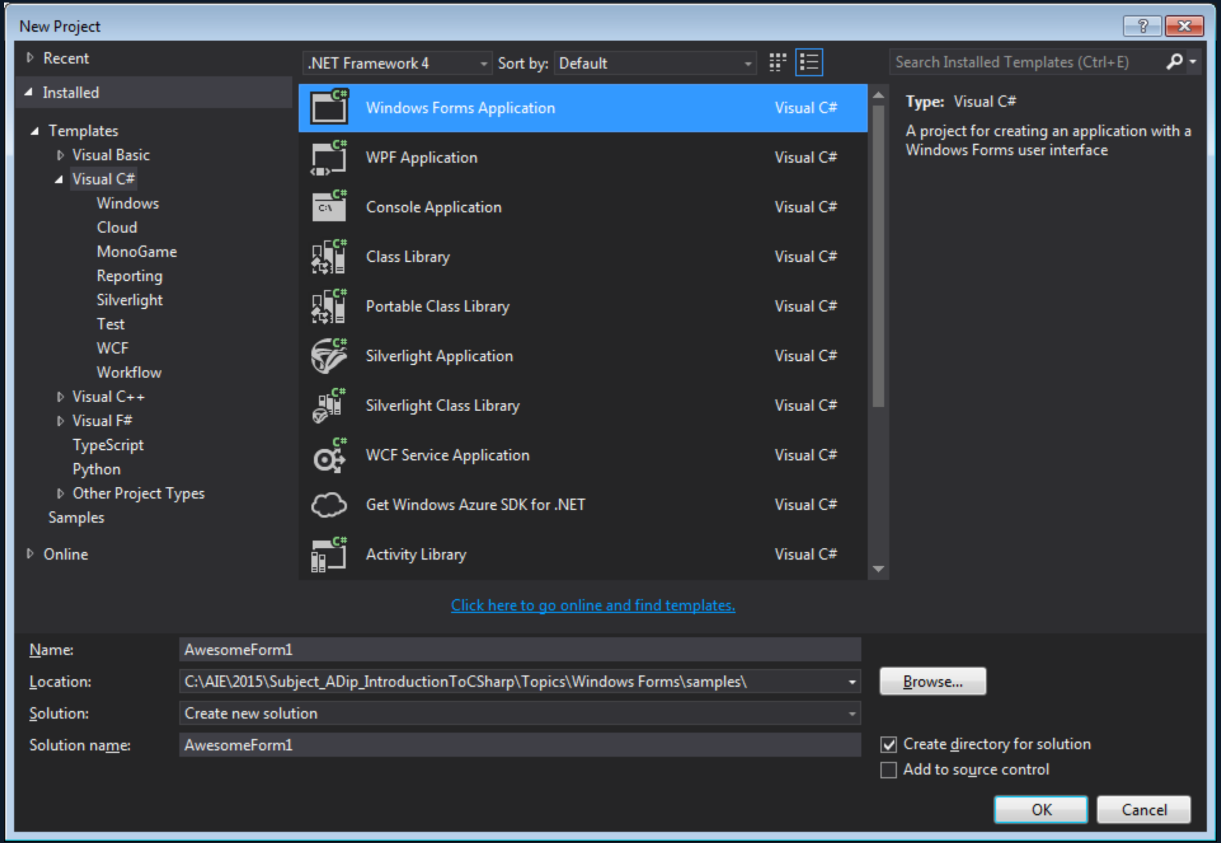
Windows Forms Overview

- Develop ‘smart clients’
  - Easy to deploy and update, work connected or disconnected from internet
  - Access local resources in a more secure manner
- *Form*: visual surface to display information to the user
- Control: a discrete UI element displaying data or accepting input
  - Added to forms
  - Respond to user actions

Windows Forms Overview

- Windows Forms contains a variety of controls:
  - Text boxes, buttons, drop-down boxes, radio buttons, Web pages, and more
- Rich UI controls
  - ToolStrip and MenuStrip
- Supports creating custom controls
- VS includes a drag-and-drop

Creating a New Form



New Project

.NET Framework 4

Sort by: Default

☑ Windows Forms Application — Visual C#

Type: Visual C#

A project for creating an application with a Windows Forms user interface

Name: AwesomeForm1

Location: C:\AIE\2015\Subject_ADip_IntroductionToCSharp\Topics\Windows Forms\samples\

Solution: Create new solution

Solution name: AwesomeForm1

☑ Create directory for solution

☐ Add to source control

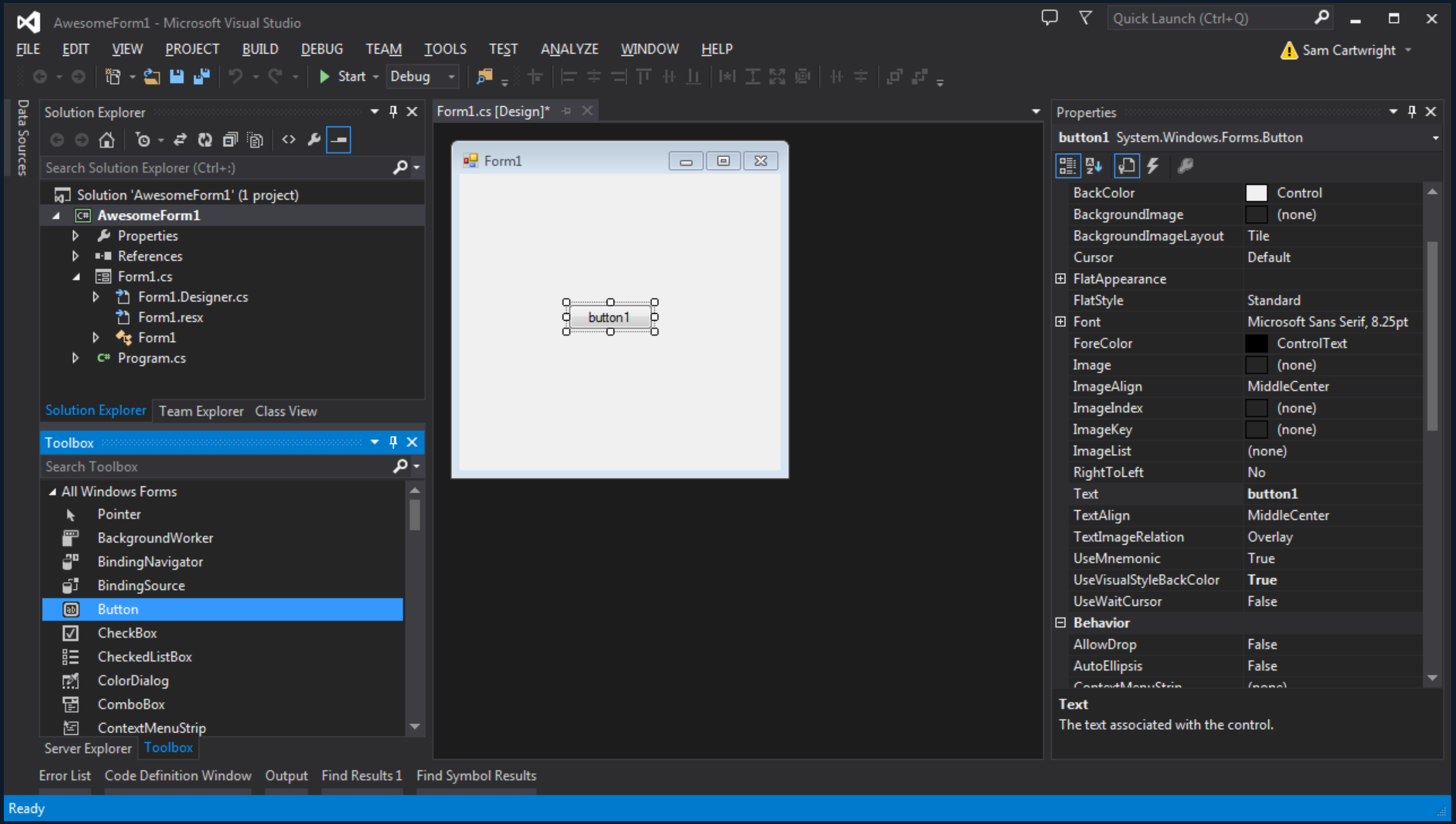
Recent
Installed
Templates
Visual Basic
Visual C#
Windows
Cloud
MonoGame
Reporting
Silverlight
Test
WCF
Workflow
Visual C++
Visual F#
TypeScript
Python
Other Project Types
Samples
Online
Search Installed Templates (Ctrl+E)
Windows Forms Application Visual C#
WPF Application Visual C#
Console Application Visual C#
Class Library Visual C#
Portable Class Library Visual C#
Silverlight Application Visual C#
Silverlight Class Library Visual C#
WCF Service Application Visual C#
Get Windows Azure SDK for .NET Visual C#
Activity Library Visual C#
Click here to go online and find templates.
Browse...
OK
Cancel

Canberra Institute of Technology

aie

SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

Creating a New Form



button1 System.Windows.Forms.Button

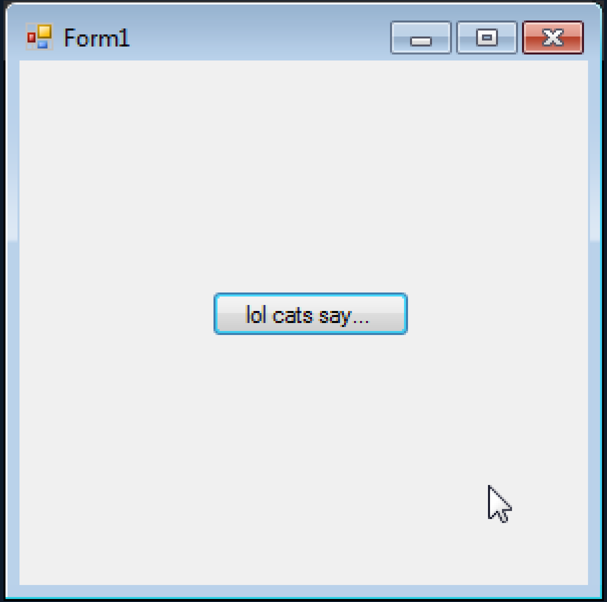
BackColor	Control
BackgroundImage	(none)
BackgroundImageLayout	Tile
Cursor	Default
⊞ FlatAppearance	
FlatStyle	Standard
⊞ Font	Microsoft Sans Serif, 8.25pt
ForeColor	ControlText
Image	(none)
ImageAlign	MiddleCenter
ImageIndex	(none)
ImageKey	(none)
ImageList	(none)
RightToLeft	No
Text	**button1**
TextAlign	MiddleCenter
TextImageRelation	Overlay
UseMnemonic	True
UseVisualStyleBackColor	**True**
UseWaitCursor	False
⊟ **Behavior**	
AllowDrop	False
AutoEllipsis	False
ContextMenuStrip	(none)

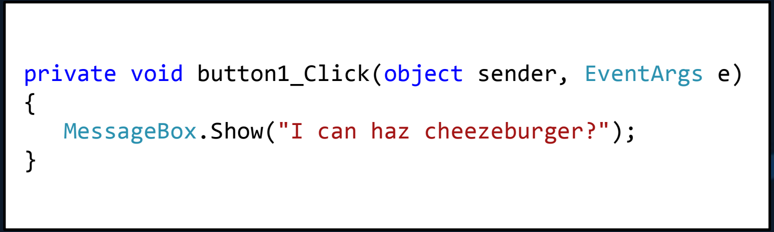
Text
The text associated with the control.

Canberra Institute of Technology

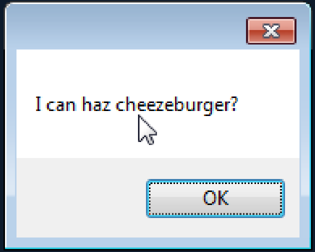
aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

Creating a New Form





```
private void button1_Click(object sender, EventArgs e)
{
   MessageBox.Show("I can haz cheezeburger?");
}
```



I can haz cheezeburger?

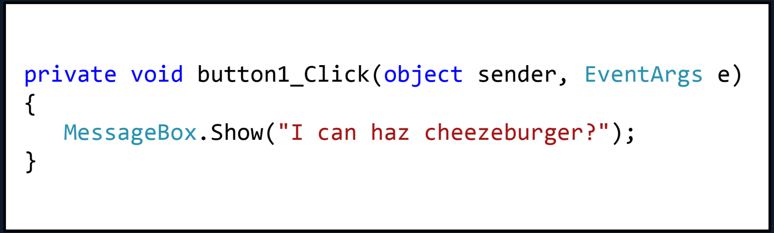
Event Driven Programming

- Traditional – the program itself fully dictates what code is executed and when
- C# incorporates an event-driven programming model
  - Execute code in response to an event
  - No pre-determined order of events
  - Many events occur in conjunction with other events

Creating Event Handlers in Windows Forms

- Event Handler: function determining what actions are performed when an event occurs
  - For example, a button click
- Events can be assigned to multiple handlers
- The methods that handle particular events can be changed dynamically
- Use the Forms Designer to create event handlers

Creating Event Handlers in Windows Forms



```
private void button1_Click(object sender, EventArgs e)
{
   MessageBox.Show("I can haz cheezeburger?");
}
```

- Event handlers provide 2 parameters
  - sender: reference to the object that raised the event
  - e: an object specific to the event being handled
- Obtain information such as location of mouse (mouse events) or data transferred (drag-and-drop events)

Windows Forms (Basic) Controls

- Button: Presents a standard button that the user can click to perform actions
- CheckBox: Indicates whether a condition is on or off
- ColorDialog: Select a colour from a palette in a pre-configured dialog box
- ComboBox: Displays data in a drop-down combo box
- DateTimePicker: Select from a list of dates or times
- FileDialog: Functionality for file dialog boxes
- Label: Displays text that cannot be edited by the user

Windows Forms (Basic) Controls

- ListView: Displays a list of items with icons, in the manner of Windows Explorer
- PictureBox: Displays graphics in BMP, GIF, JPEG, metafile, or icon format
- ProgressBar: Graphically indicates progress of an action
- RichTextBox: Enter, display, and manipulate text with formatting
- ToolBar: Displays menus and bitmapped buttons that activate commands
- WebBrowser: Hosts Web pages and provides Web browsing capabilities

https://msdn.microsoft.com/en-us/library/3xdhey7w(v=vs.110).aspx

Canberra Institute of Technology

aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

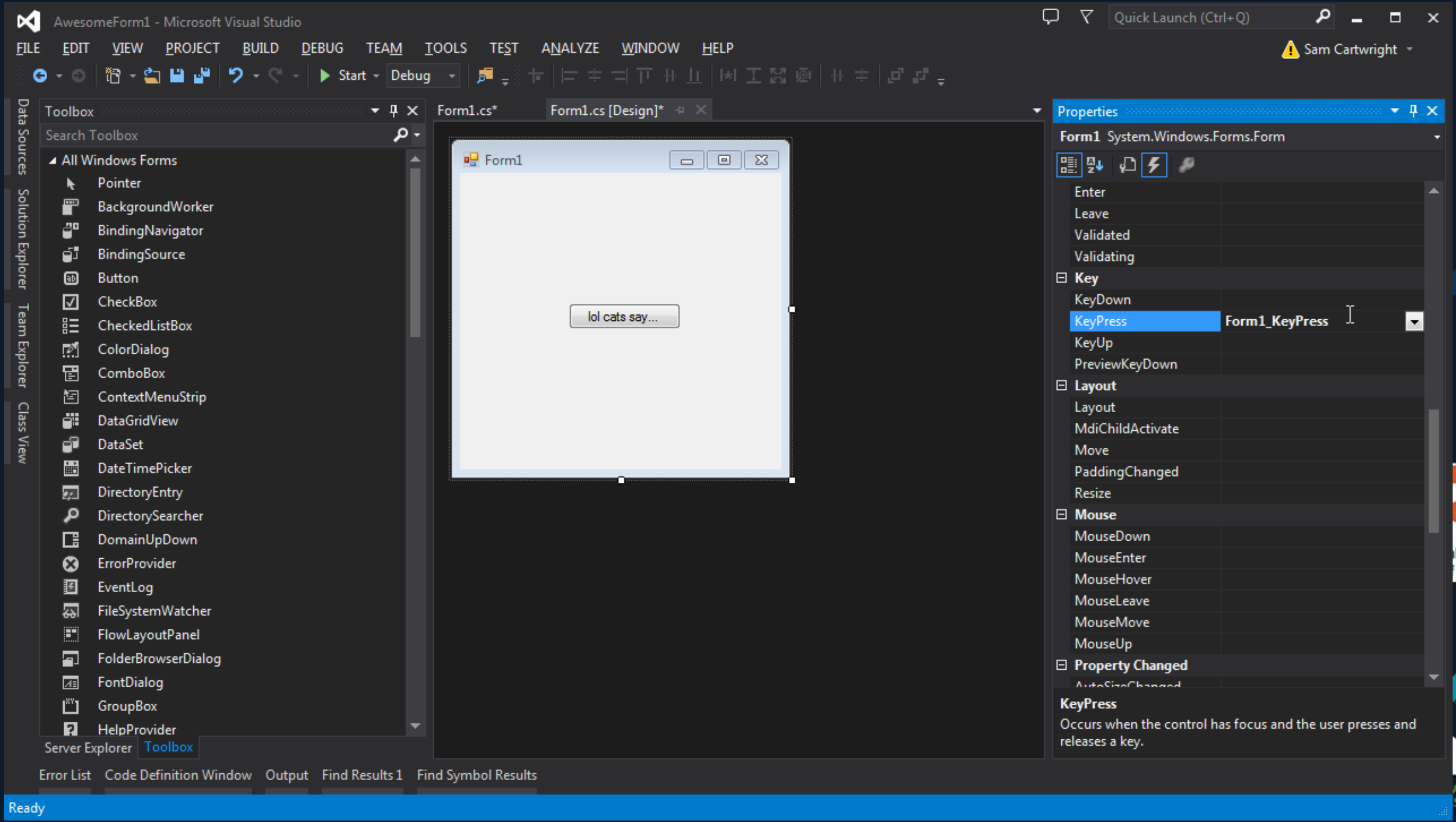
User Input

- Sent to applications as Windows messages
- Processed by a series of overridable methods
- These methods raise events to be handled by the Windows Form Application
- You can query the event to get information about the mouse/keyboard input

Keyboard and Mouse Events

- All form controls inherit a set of events for mouse/keyboard input
  - For example, KeyPress event to determine the character code of a key pressed, or
  - MouseClick event to determine the location of a mouse click

KeyPress Event

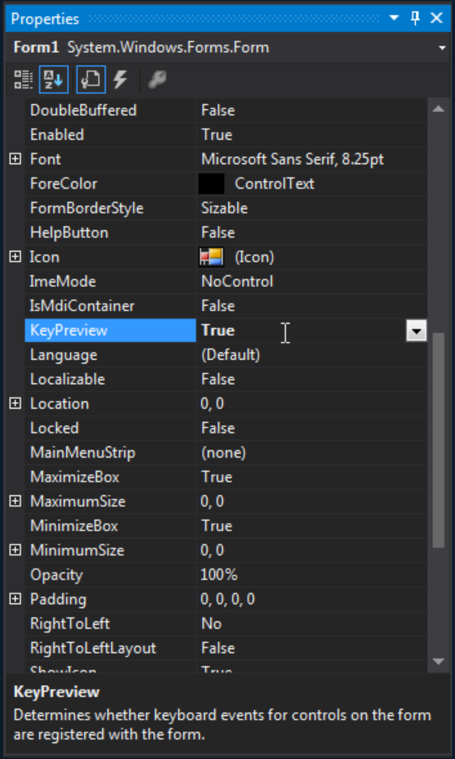


KeyPress Form1_KeyPress

KeyPress

Occurs when the control has focus and the user presses and releases a key.

KeyPress Event

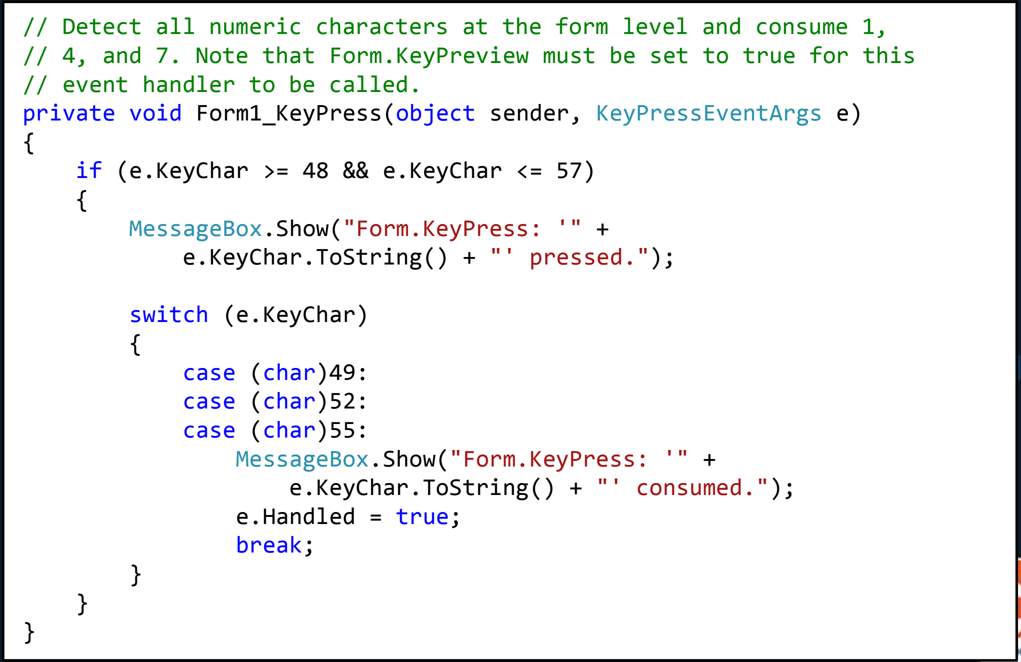


Properties

Form1 System.Windows.Forms.Form

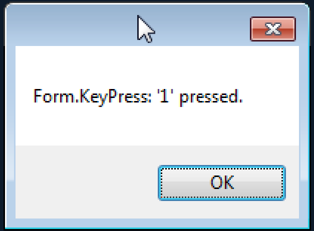
DoubleBuffered	False
Enabled	True
⊞ Font	Microsoft Sans Serif, 8.25pt
ForeColor	■ ControlText
FormBorderStyle	Sizable
HelpButton	False
⊞ Icon	(Icon)
ImeMode	NoControl
IsMdiContainer	False
KeyPreview	**True**
Language	(Default)
Localizable	False
⊞ Location	0, 0
Locked	False
MainMenuStrip	(none)
MaximizeBox	True
⊞ MaximumSize	0, 0
MinimizeBox	True
⊞ MinimumSize	0, 0
Opacity	100%
⊞ Padding	0, 0, 0, 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True

KeyPreview
Determines whether keyboard events for controls on the form are registered with the form.

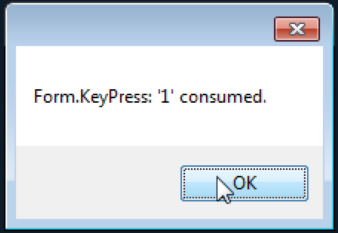


```
// Detect all numeric characters at the form level and consume 1,
// 4, and 7. Note that Form.KeyPreview must be set to true for this
// event handler to be called.
private void Form1_KeyPress(object sender, KeyPressEventArgs e)
{
    if (e.KeyChar >= 48 && e.KeyChar <= 57)
    {
        MessageBox.Show("Form.KeyPress: '" +
            e.KeyChar.ToString() + "' pressed.");

        switch (e.KeyChar)
        {
            case (char)49:
            case (char)52:
            case (char)55:
                MessageBox.Show("Form.KeyPress: '" +
                    e.KeyChar.ToString() + "' consumed.");
                e.Handled = true;
                break;
        }
    }
}
```



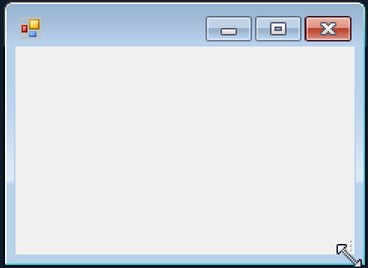
Form.KeyPress: '1' pressed.



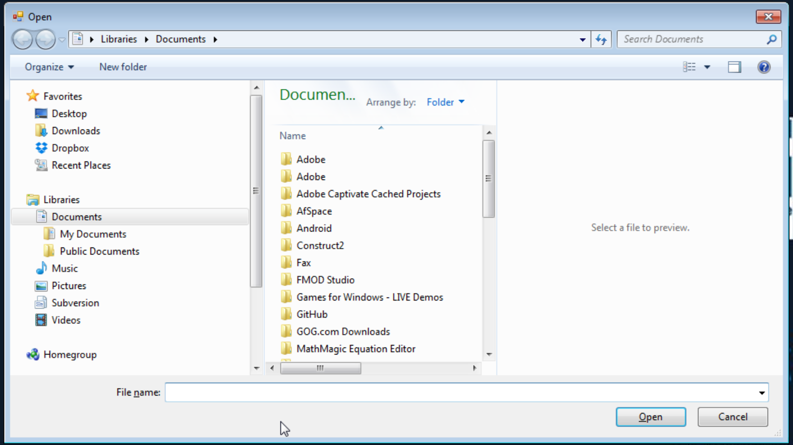
Form.KeyPress: '1' consumed.

Dialog Boxes

```
private void button2_Click(object sender, EventArgs e)
{
    Form dlg1 = new Form();
    dlg1.ShowDialog();
}
```



```
private void button2_Click(object sender, EventArgs e)
{
    OpenFileDialog dlg1 = new OpenFileDialog();
    dlg1.ShowDialog();
}
```



Summary

- Windows Forms contains a variety of rich UI controls
- Easily drag-and-drop using the Forms Designer
- Set properties to change the appearance of controls
- Add event handlers to respond to user input
- Create custom dialogs or use existing ones