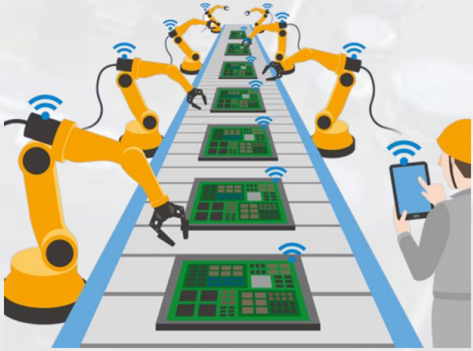
# Week #02 – Introduction to OOP

# INC382 Capstone Project

for

# Automation Engineers



**Santi Nuratch., Ph.D.**

**Embedded Computing and Control Lab. @ INC-KMUTT**

santi.inc.kmutt@gmail.com, santi.nur@kmutt.ac.th

Department of Control System and Instrumentation Engineering,
King Mongkut's University of Technology Thonburi, **KMUTT**

# Objectives

- C# Programming on Visual Studio 2017
- Objects in Window Form Application
- Existing Objects and Dynamic Objects Creation
- Create the first Class and its Property and Method

Practice for Knowledge Acquisition

Set Goal
Plan
Learn
Show
Reflect

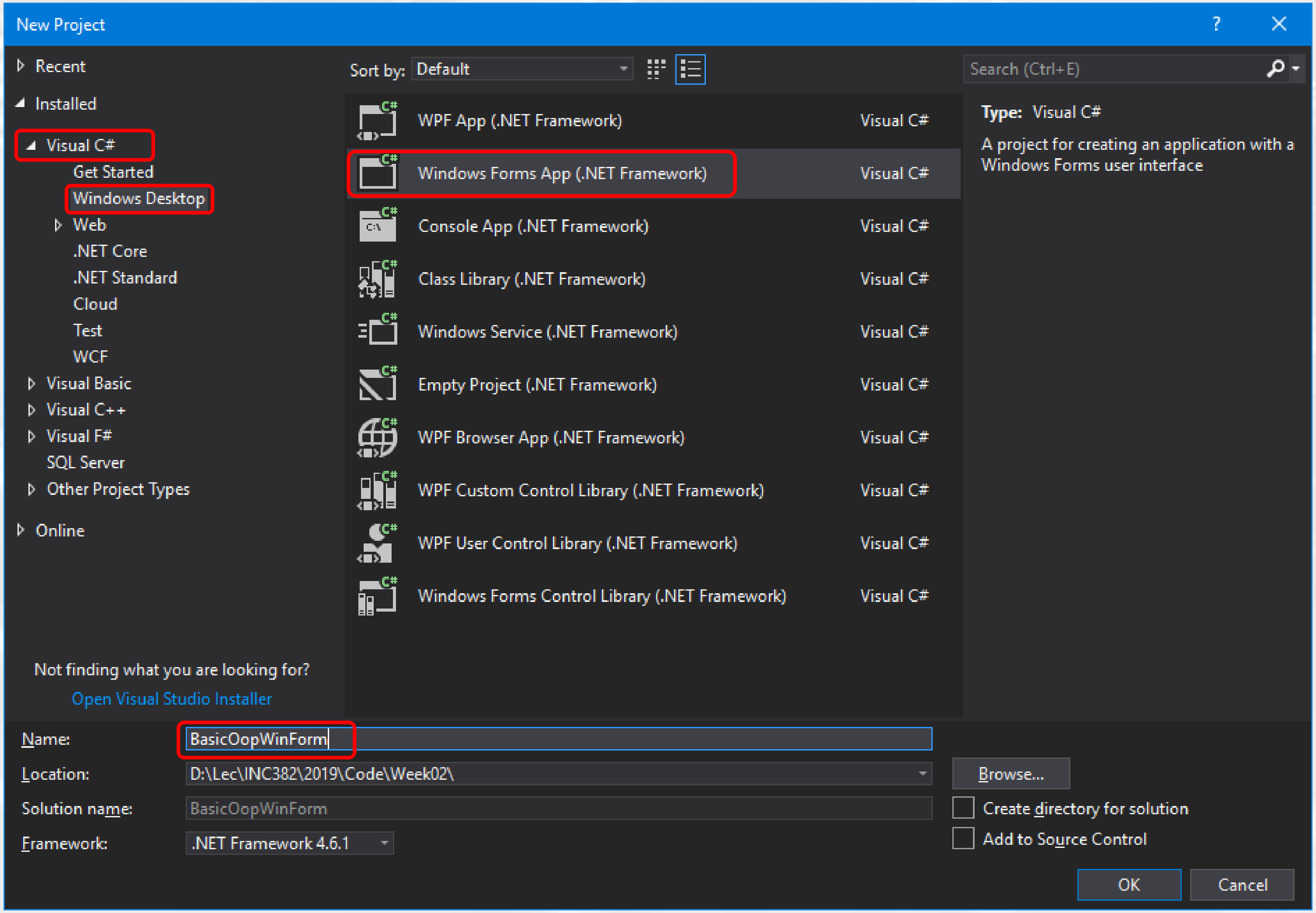
www.c-sharpcorner.com

www.tutorialspoint.com

When we practice something, we are involved in the deliberate repetition of a process with the intention of reaching a specific goal

# New Project

1) Create a new Project, **Visual C# | Windows Desktop | Windows Forms App**. Give it a name **BasicOopWinForm**



New Project

Recent

Installed

- Visual C#
  - Get Started
  - Windows Desktop
  - Web
  - .NET Core
  - .NET Standard
  - Cloud
  - Test
  - WCF
- Visual Basic
- Visual C++
- Visual F#
- SQL Server
- Other Project Types

Online

Sort by: Default

Search (Ctrl+E)

| | |
|---|---|
| WPF App (.NET Framework) | Visual C# |
| Windows Forms App (.NET Framework) | Visual C# |
| Console App (.NET Framework) | Visual C# |
| Class Library (.NET Framework) | Visual C# |
| Windows Service (.NET Framework) | Visual C# |
| Empty Project (.NET Framework) | Visual C# |
| WPF Browser App (.NET Framework) | Visual C# |
| WPF Custom Control Library (.NET Framework) | Visual C# |
| WPF User Control Library (.NET Framework) | Visual C# |
| Windows Forms Control Library (.NET Framework) | Visual C# |

**Type:** Visual C#

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

Not finding what you are looking for?

Open Visual Studio Installer

Name: BasicOopWinForm

Location: D:\Lec\INC382\2019\Code\Week02\

Solution name: BasicOopWinForm

Framework: .NET Framework 4.6.1

Browse...

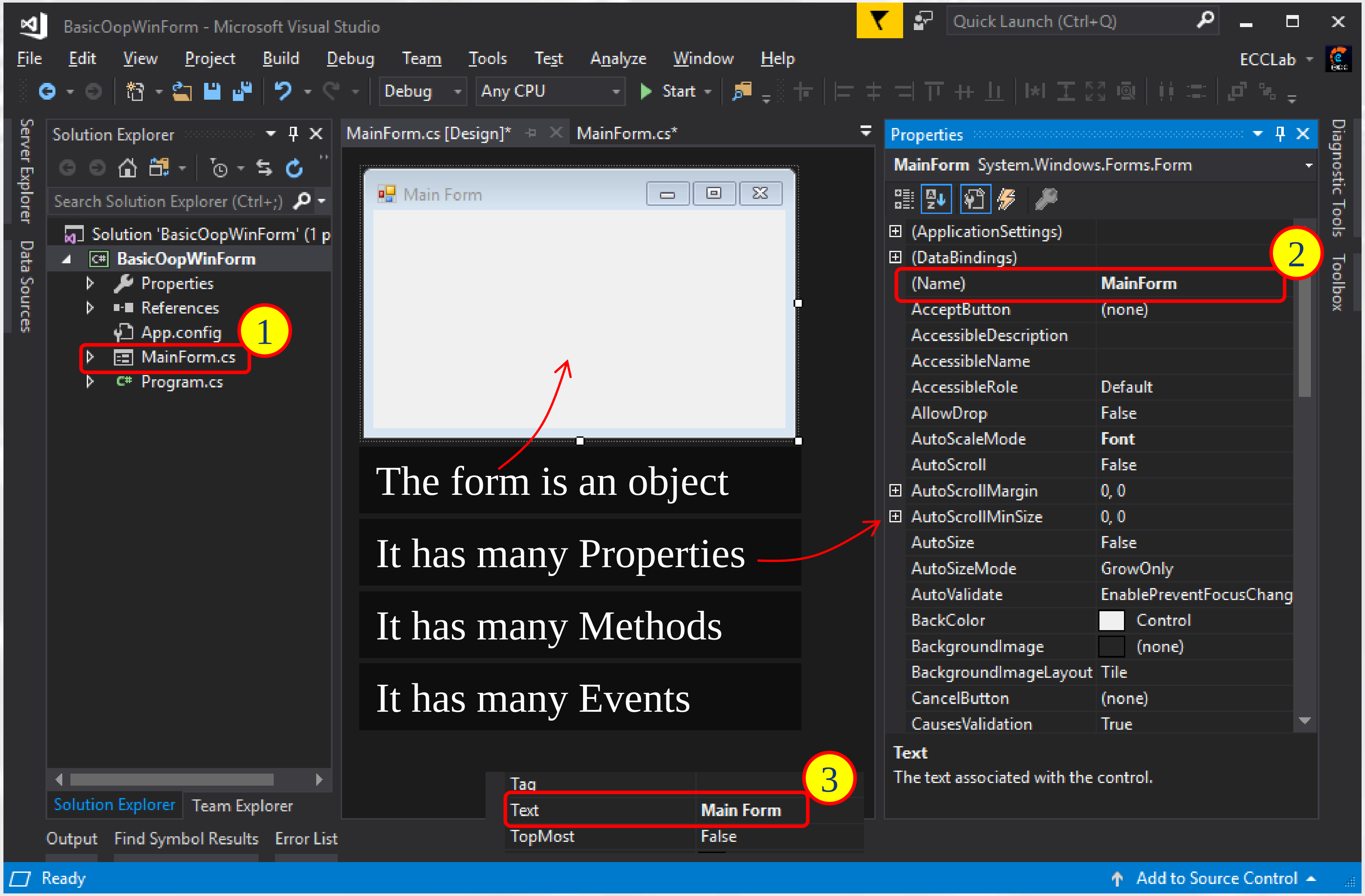
□ Create directory for solution

□ Add to Source Control

OK Cancel

# New Project

2) Rename Form1.cs to **MainForm.cs**, change the form name to **MainForm** and change the form text (caption/title) to **Main Form**



The form is an object

It has many Properties

It has many Methods

It has many Events

MainForm System.Windows.Forms.Form

| | |
|---|---|
| ⊞ (ApplicationSettings) | |
| ⊞ (DataBindings) | |
| (Name) | **MainForm** |
| AcceptButton | (none) |
| AccessibleDescription | |
| AccessibleName | |
| AccessibleRole | Default |
| AllowDrop | False |
| AutoScaleMode | **Font** |
| AutoScroll | False |
| ⊞ AutoScrollMargin | 0, 0 |
| ⊞ AutoScrollMinSize | 0, 0 |
| AutoSize | False |
| AutoSizeMode | GrowOnly |
| AutoValidate | EnablePreventFocusChang |
| BackColor | □ Control |
| BackgroundImage | ■ (none) |
| BackgroundImageLayout | Tile |
| CancelButton | (none) |
| CausesValidation | True |

**Text**
The text associated with the control.

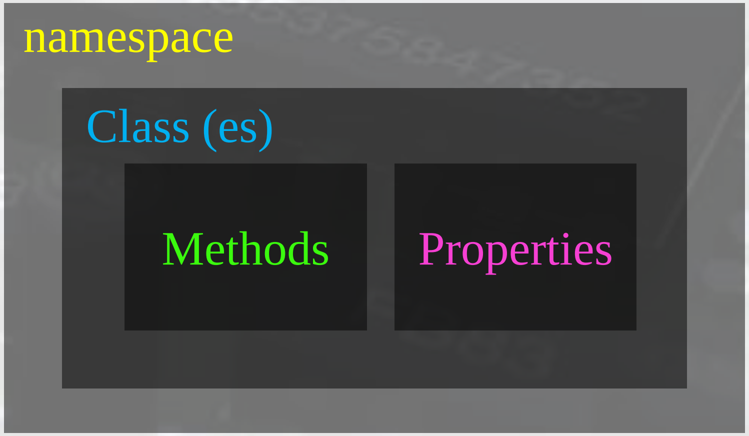
| | |
|---|---|
| Tag | |
| Text | **Main Form** |
| TopMost | False |

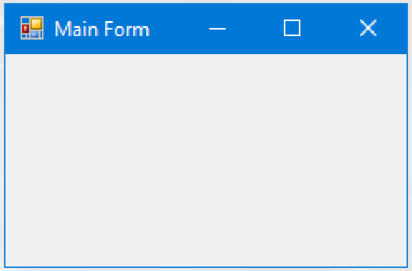
# Namespace and Class

3) Double-click on the **Program.cs** and check the **namespace**, **class** name and **Main** method

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace BasicOopWinForm
{
    static class MainForm
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new MainForm());
        }
    }
}
```





The **MainForm** object is created by **new** keyword and is executed by the **Application**

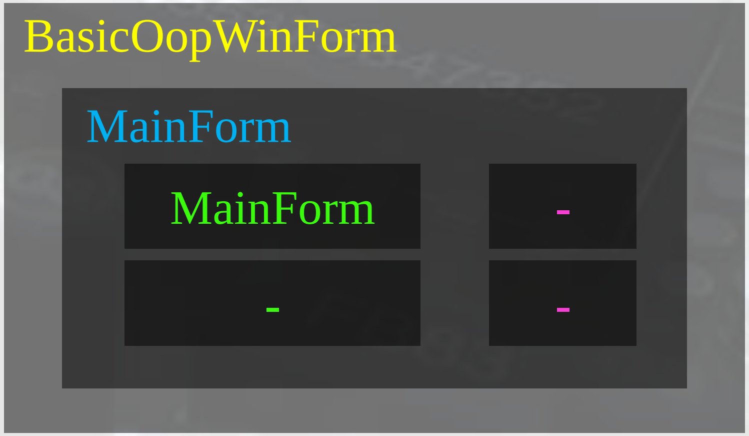
# Constructor

4) Right-click on the **MainForm** and click **View Code** (F7)

```
using System.Windows.Forms;

namespace BasicOopWinForm
{
    public partial class MainForm : Form
    {
        public MyForm()
        {
            InitializeComponent();
        }
    }
}
```

Constructor



The method that has the same name with the class name and no return is called “**Constructor**”

The **Constructor** ( MainForm( ) ) will be executed immediately after the object (MainForm) is created

# Existing Classes and Instances

5) Create a new object of **Panel** class called **RedBox** and add to the **MainForm**

```
using System.Windows.Forms;

namespace BasicOopWinForm
{
    public partial class MainForm : Form
    {

        public Panel RedBox;

        public MainForm()
        {
            InitializeComponent();

            RedBox = new Panel();
            RedBox.BackColor = Color.Red;

            this.Controls.Add(RedBox);
        }
    }
}
```

**Panel** is a class name

Declare an object, the RedBox

Create new instance(object) of Panel

Change object property

Add the object into the MainForm

this keyword is a special variable used as reference of this instance (object) of this class (MainForm)

# Event and Handler

6) Add **Click** event to the **RedBox** object and add event **Handler**, the **RedBox_Click( )**

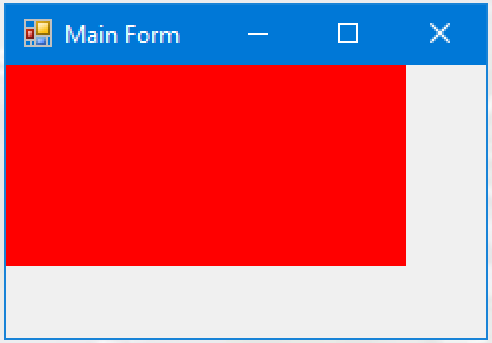
```
namespace BasicOopWinForm
{
    public partial class MainForm : Form
    {
        public Panel RedBox;

        public MainForm()
        {
            InitializeComponent();
            RedBox = new Panel();
            RedBox.BackColor = Color.Red;

            this.Controls.Add(RedBox);

            RedBox.Click += RedBox_Click;
        }

        private void RedBox_Click(object sender, EventArgs e)
        {
            RedBox.BackColor =
                          RedBox.BackColor == Color.Red ? Color.Lime : Color.Red;
        }
    }
}
```



Add click event to the RedBox

RedBox_Click is event handler

# Method and Property

7) Modify the **RedBox_Click( )**, then add the **ChangeSize( )** and **ChangePosition( )**

```
private void RedBox_Click(object sender, EventArgs e)
{
    Random random = new Random();

    int width  = 50 + random.Next(50);
    int height = 50 + random.Next(50);
    this.ChangeSize(width, height);

    int left = random.Next(this.Width  - width);
    int top  = random.Next(this.Height - height);
    ChangePosition(left, top);
}

private void ChangeSize(int width, int height)
{
    RedBox.Width  = width;
    RedBox.Height = height;
}
private void ChangePosition(int left, int top)
{
    RedBox.Left = left;
    RedBox.Top  = top;
}
```

**Random is built-in static class**

Randomly generate **width** and **height** values

Call the **ChangeSize()** method

Randomly generate **left** and **top** values

Call the **ChangePosition()** method

Change **Width** and **Height** properties of the RedBox object

Change **Left** and **Top** properties of the RedBox object

# Object and Sender

8) Add the class properties and modify the MainForm( )

```
public partial class MainForm : Form
{
    private Panel RedBox;

    private int NumberOfBoxes = 5;
    private int BoxWidth  = 20;
    private int BoxHeight = 20;
    private int BoxMargin = 10;


    public MainForm()
    {
        InitializeComponent();

        for(int i=0; i<this.NumberOfBoxes; i++)
        {
            Panel box  = new Panel();
            box.Width  = this.BoxWidth;
            box.Height = this.BoxHeight;
            box.Left   = i * (this.BoxWidth + this.BoxMargin);
            box.Top    = 20;
            box.BackColor   = Color.Red;
            box.BorderStyle = BorderStyle.FixedSingle;
            this.Controls.Add(box);
            box.Click += Box_Click;
        }
    }
```

All boxes share the Box_Click

# Object and Sender

9) Rename the RedBox_Click( ) to Box_Click( ), then modify the ChangeSize( ) and ChangePosition( )

```
        private void Box_Click(object sender, EventArgs e)
        {
            Panel clickedBox = (Panel)sender;

            Random random = new Random();

            int width  = clickedBox.Width  + 1;
            int height = clickedBox.Height + 1;
            this.ChangeSize(clickedBox, width, height);

            int left = clickedBox.Left + 1;
            int top  = clickedBox.Top  + 1;
            this.ChangePosition(clickedBox, left, top);
        }

        private void ChangeSize(Panel box, int width, int height)
        {
            box.Width  = width;
            box.Height = height;
        }
        private void ChangePosition(Panel box, int left, int top)
        {
            box.Left = left;
            box.Top  = top;
        }
```

The sender is a box that emits this event

# It’s time to learn More and More..

Derek Banas, C#

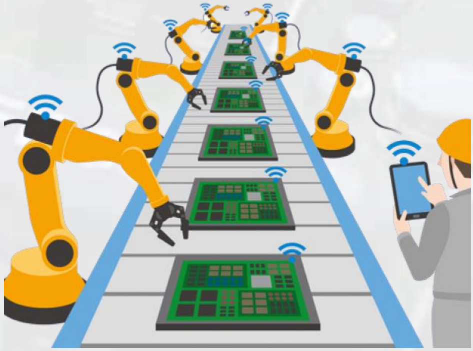
Kudvenkat, C#

www.c-sharpcorner.com

www.tutorialspoint.com

# THANK YOU!

we make computers do more.



**Santi Nuratch., Ph.D.**

**Embedded Computing and Control Lab. @ INC-KMUTT**

santi.inc.kmutt@gmail.com, santi.nur@kmutt.ac.th

Department of Control System and Instrumentation Engineering,
King Mongkut's University of Technology Thonburi, **KMUTT**