# Lecture # 5:

**Objective:**

* Windows Form Application
* Events
* Properties
* GUI Programming Techniques(Phases)
* Calculation Program Example

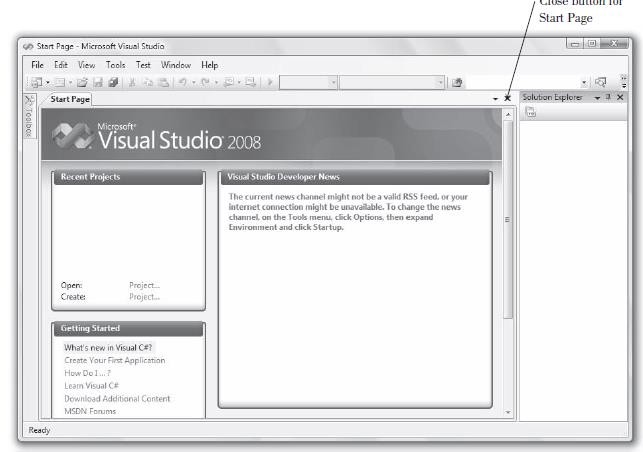
# Windows Form Application:

Sequential programming model is used by console based applications, in which application starts from a point, performs some input/output, selection or iterative operations and stops to another point.

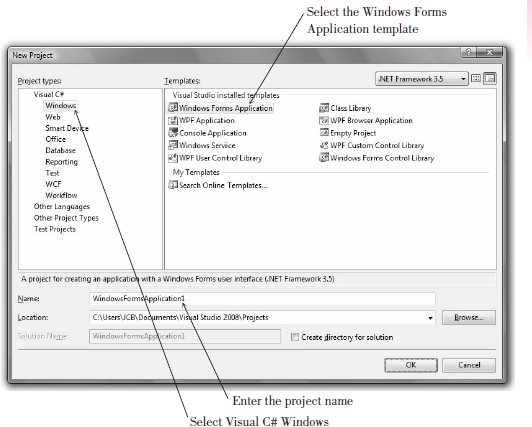
In event driven programming model, application starts from a point and then it enters a message loop. This message loop monitors different events which may occur on application, application executes different event handlers which are associated with these events.

# Introduction to Visual Studio .Net environment:

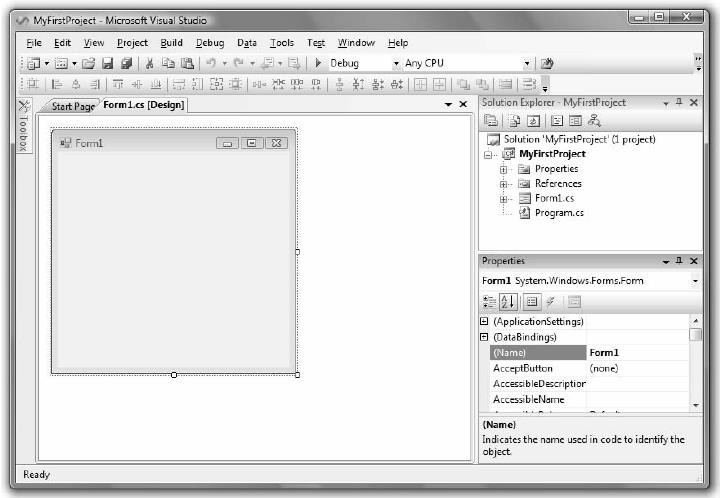
Visual Studio .Net 2008 programming environment is shown below:



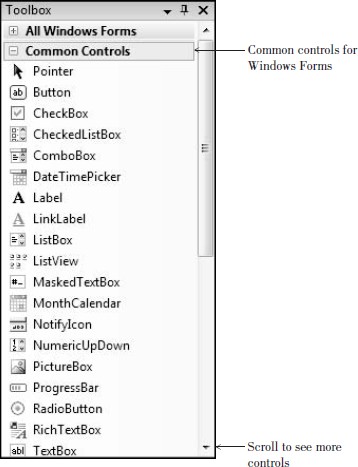
Following windows is visible when we select File  New  Project menu items from main window of visual studio .Net.



Main window of IDE of Visual Studio .Net is shown in following figure, when a template type of Windows Form Based Application is selected.



Following toolbox is used to place different user interface elements on a form for a form based application.



**GUI Programming Technique:** Programming technique for GUI based applications is based on three phases, which are described as follows:

1. Analysis Phase: Gathering information about problem statement and checking solution which is already exisists and planning which feature should be extended or changed.
2. Designing Phase: Preparation of Object Property Sheet and Object Activity Sheet documents for application.
3. Coding/Programming Phase: Development of GUI application in Visual Studio .Net by designing GUI layout from planing phase, then setting properties of GUI elements of application using Object Property Sheet. Further programming for event handler functions is done by using Oject Activity Sheet.

**Object Property Sheet:** This sheet contains information about properties and their values for different GUI elements for a given GUI screen.

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr #** | **Control Name** | **Property** | **Value** |
| **1** | Label | Text | Enter 1st value |

|  |  |
| --- | --- |
| Name | lblVal1 |
|  | |

**Object Activity Sheet:** This sheet represents pseudo code of event handler functions which are associated with an event which is caused on a user interface element.

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr #** | **Control Name** | **Event** | **Pseudo-Code** |
| **1** | Button | Click | * Validate all fields that is no will will be left empty. * Show message validate successfully or not. |
|  | | | |
|  | | | |

# Calculation Program:

Create an application which inputs two integer type values from user in a text box. There are four button named as ADD, SUB, MUL, DIV which performs addition of two number, subtraction of two numbers, multiplication of two numbers and division of two numbers respectively. And show result in another textbox of read-only property as TRUE. There is another button named as CLEAR which clears all text box values, and EXIT button closes the application.

Solution:

using System;

using System.Collections.Generic; using System.ComponentModel; using System.Data;

using System.Drawing; using System.Linq; using System.Text;

using System.Windows.Forms;

namespace calculationApplication

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void btnAdd\_Click(object sender, EventArgs e)

{

if (txtVal1.Text == "")

{

MessageBox.Show("Enter value 1"); txtVal1.Focus();

}

else if (txtVal2.Text == "")

{

MessageBox.Show("Enter 2nd value"); txtVal2.Focus();

}

else

{

int v1, v2, res;

v1 = int.Parse(txtVal1.Text); v2 = int.Parse(txtVal2.Text); res = v1 + v2;

txtRes.Text = res.ToString();

}

}

private void btnSub\_Click(object sender, EventArgs e)

{

if (txtVal1.Text == "")

{

MessageBox.Show("Enter value 1"); txtVal1.Focus();

}

else if (txtVal2.Text == "")

{

MessageBox.Show("Enter 2nd value"); txtVal2.Focus();

}

else

{

int v1, v2, res;

v1 = int.Parse(txtVal1.Text); v2 = int.Parse(txtVal2.Text); res = v1 - v2;

txtRes.Text = res.ToString();

}

}

private void btnMul\_Click(object sender, EventArgs e)

{

if (txtVal1.Text == "")

{

MessageBox.Show("Enter value 1"); txtVal1.Focus();

}

else if (txtVal2.Text == "")

{

MessageBox.Show("Enter 2nd value"); txtVal2.Focus();

}

else

{

int v1, v2, res;

v1 = int.Parse(txtVal1.Text); v2 = int.Parse(txtVal2.Text); res = v1 \* v2;

txtRes.Text = res.ToString();

}

}

private void btnDiv\_Click(object sender, EventArgs e)

{

if (txtVal1.Text == "")

{

MessageBox.Show("Enter value 1"); txtVal1.Focus();

}

else if (txtVal2.Text == "")

{

MessageBox.Show("Enter 2nd value"); txtVal2.Focus();

}

else if (int.Parse(txtVal2.Text) == 0)

{

MessageBox.Show("Divide by zero is not allowed"); txtVal2.Focus();

}

else

{

int v1, v2, res;

v1 = int.Parse(txtVal1.Text); v2 = int.Parse(txtVal2.Text); res = v1 / v2;

txtRes.Text = res.ToString();

}

}

private void btnClear\_Click(object sender, EventArgs e)

{

txtVal1.Text = string.Empty; txtVal2.Text = string.Empty; txtRes.Text = string.Empty;

}

private void btnExit\_Click(object sender, EventArgs e)

{

Application.Exit();

}

}

}