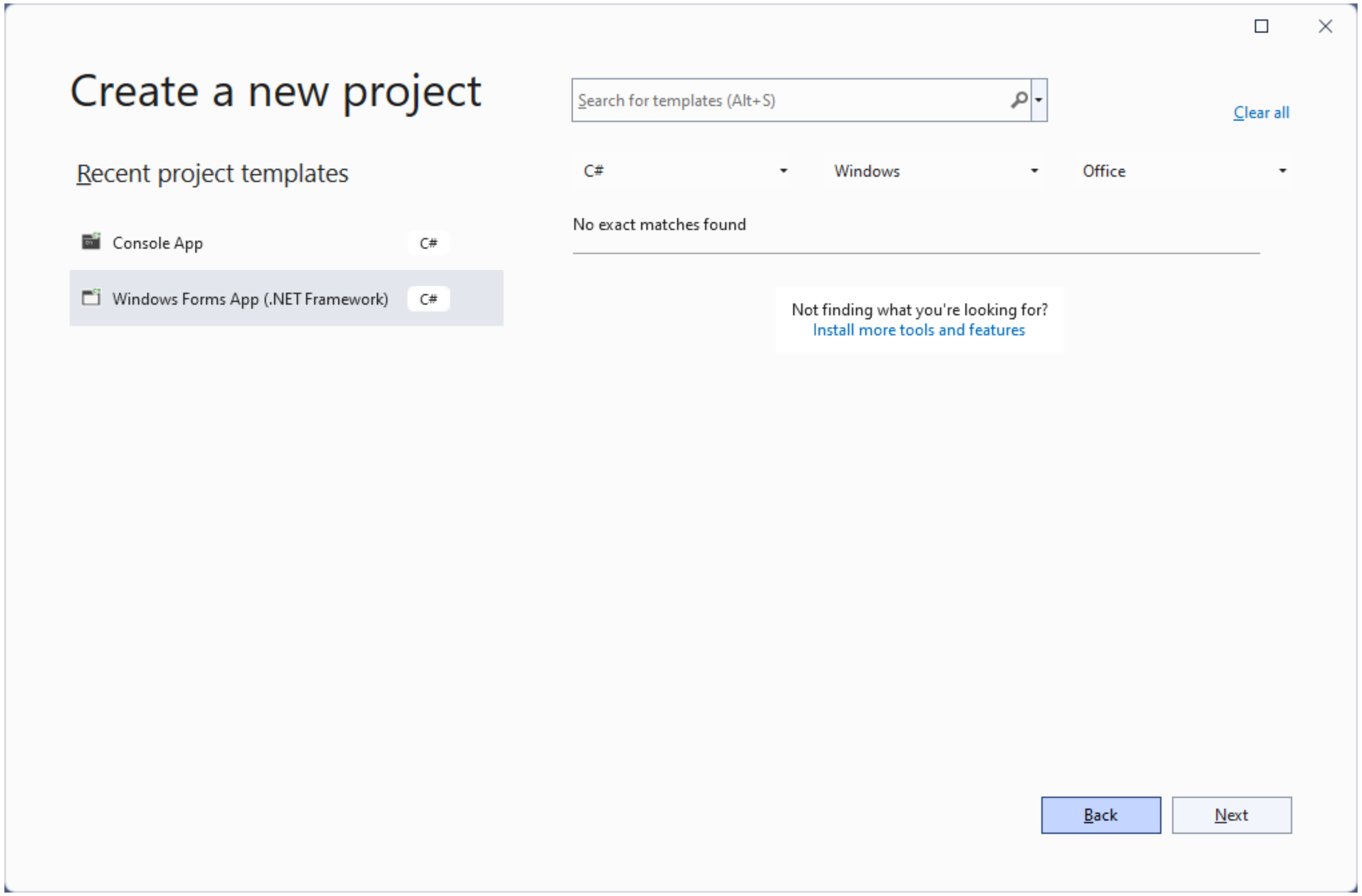
HNDIT1012 Visual Application Programming

Week 2

Windows Application - Step 1



Create a new project

Search for templates (Alt+S)

Clear all

Recent project templates

C# | Windows | Office

Console App C#

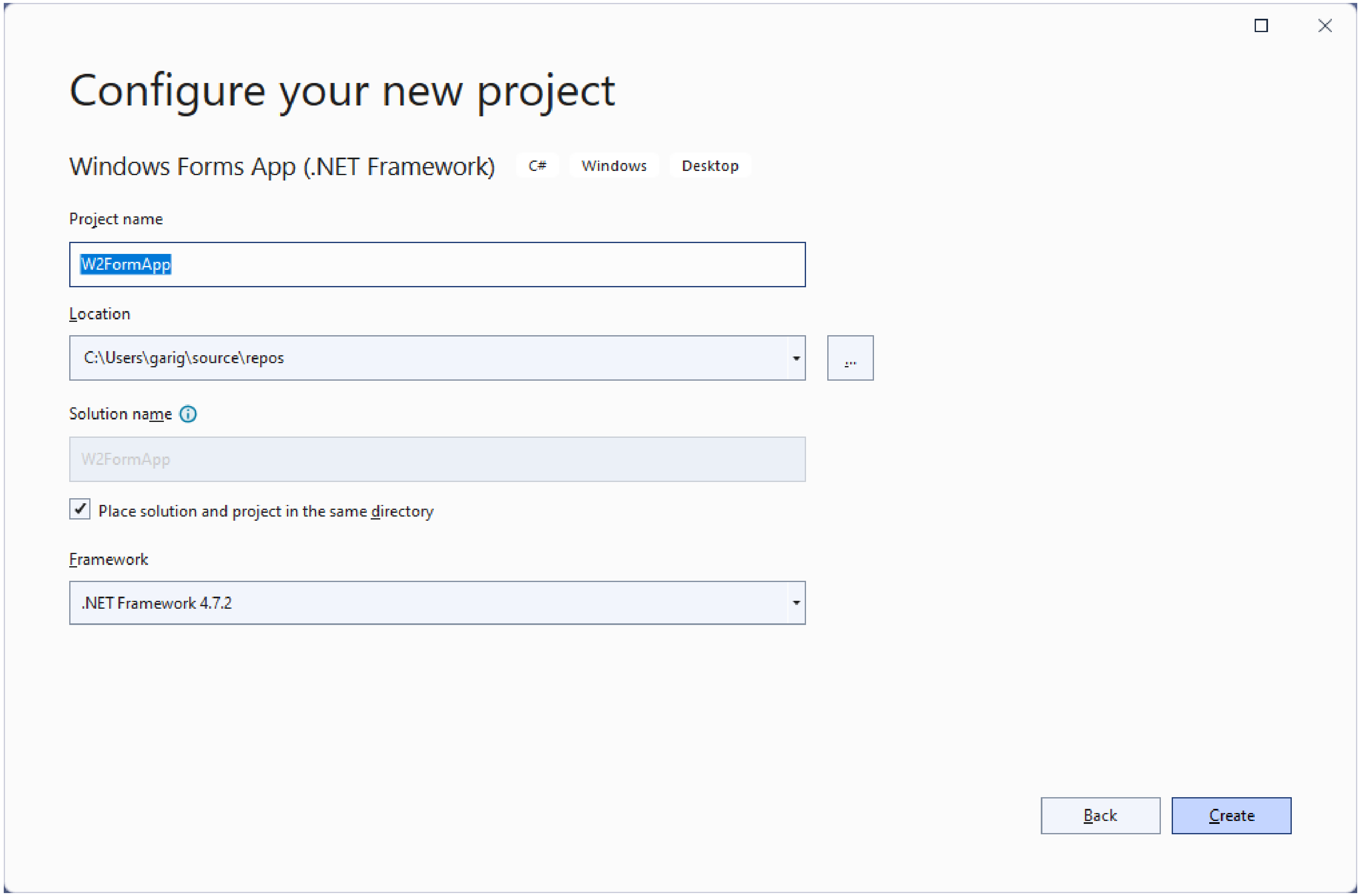
Windows Forms App (.NET Framework) C#

No exact matches found

Not finding what you're looking for?
Install more tools and features

Back Next

Step 2



Configure your new project

Windows Forms App (.NET Framework)

Project name: W2FormApp

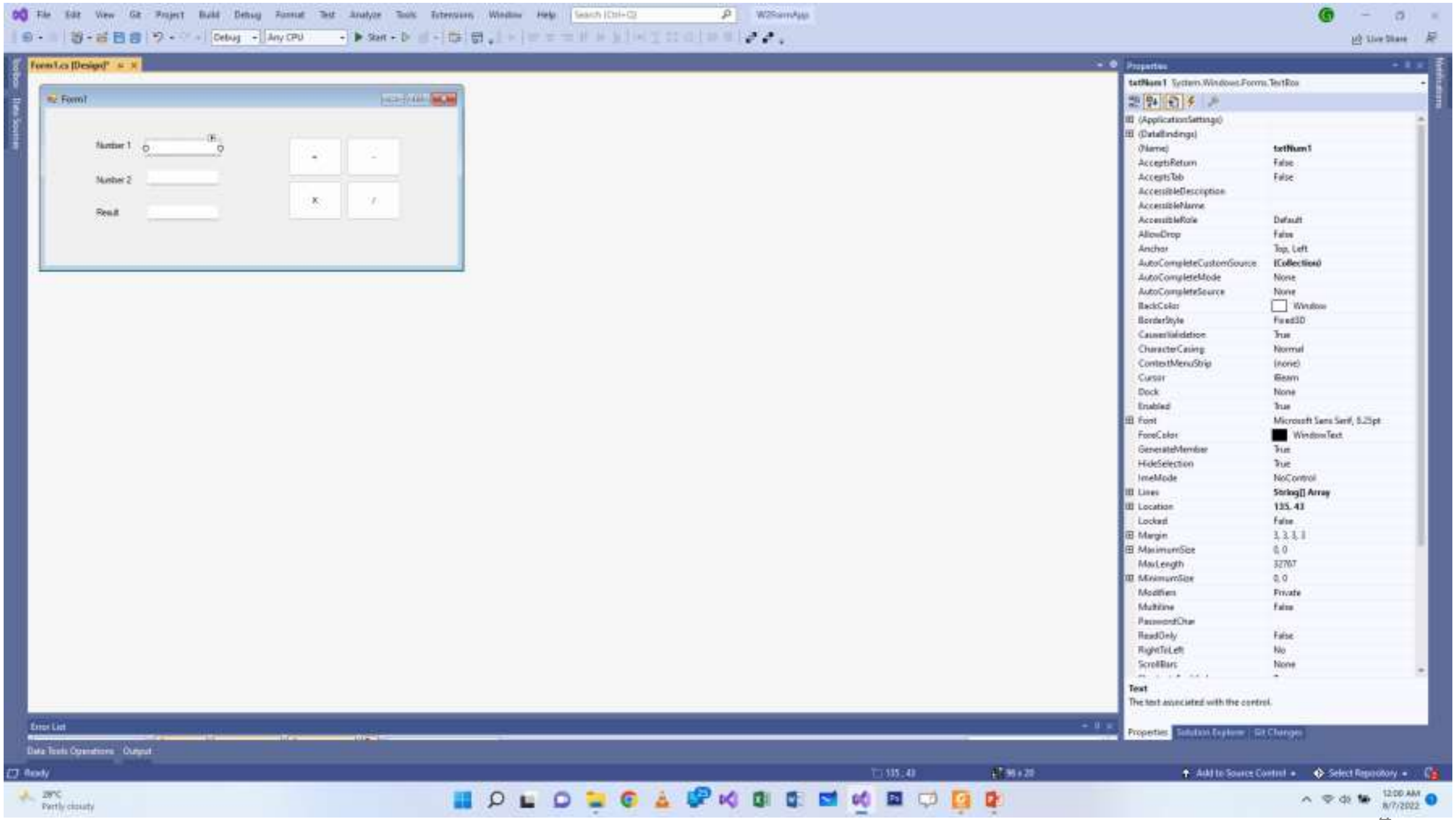
Location: C:\Users\garig\source\repos

Solution name: W2FormApp

[✓] Place solution and project in the same directory

Framework: .NET Framework 4.7.2

Designing Form – Step 3



txtNum1 System.Windows.Forms.TextBox

⊞ (ApplicationSettings)	
⊞ (DataBindings)	
(Name)	txtNum1
AcceptsReturn	False
AcceptsTab	False
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	Top, Left
AutoCompleteCustomSource	(Collection)
AutoCompleteMode	None
AutoCompleteSource	None
BackColor	⬜ Window
BorderStyle	Fixed3D
CausesValidation	True
CharacterCasing	Normal
ContextMenuStrip	(none)
Cursor	IBeam
Dock	None
Enabled	True
⊞ Font	Microsoft Sans Serif, 8.25pt
ForeColor	⬛ WindowText
GenerateMember	True
HideSelection	True
ImeMode	NoControl
⊞ Lines	String[] Array
⊞ Location	135, 43
Locked	False
⊞ Margin	3, 3, 3, 3
⊞ MaximumSize	0, 0
MaxLength	32767
⊞ MinimumSize	0, 0
Modifiers	Private
Multiline	False
PasswordChar	
ReadOnly	False
RightToLeft	No
ScrollBars	None

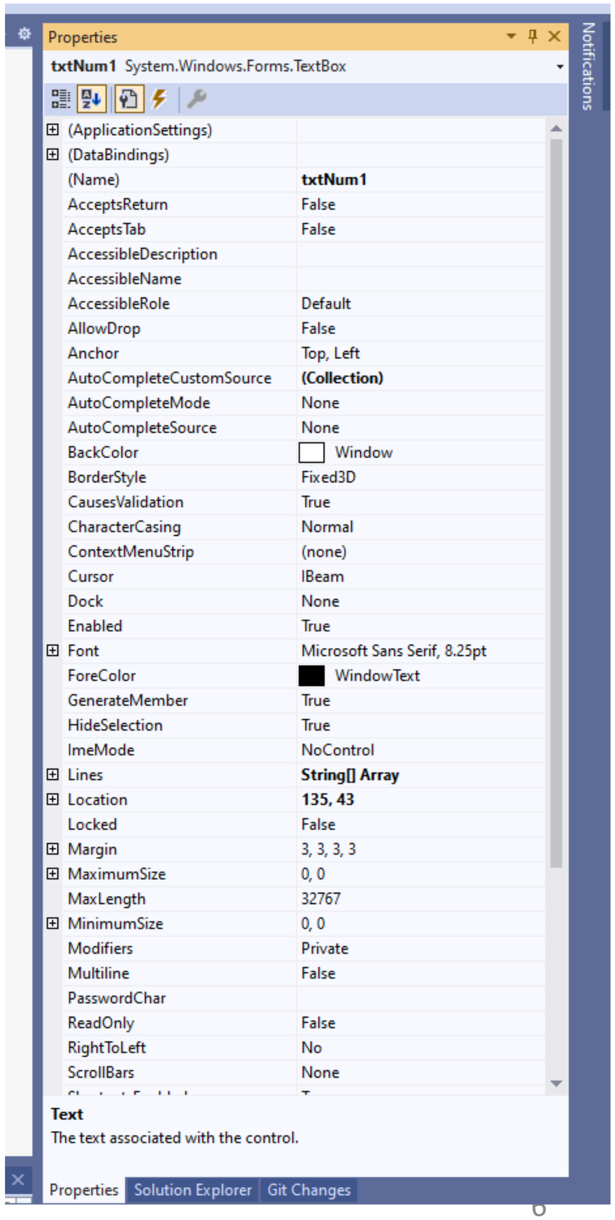
Text
The text associated with the control.

Adding Controls

- Design the form as shown above.
- The controls used in the form are:
  - TextBox :Text box controls allow entering text on a form at runtime. By default, it takes a single line of text, however, you can make it accept multiple texts and even add scroll bars to it. Let's create a text box by dragging a Text Box control from the Toolbox and dropping it on the form.
  - Button : Button control is used to perform a click event in Windows Forms, and it can be clicked by a mouse or by pressing Enter keys.
  - Label : The Label control represents a standard Windows label. It is generally used to display some informative text on the GUI which is not changed during runtime.

Property Window

- You can find Properties Window on the View menu. You can also open it by pressing F4 or by typing Properties in the search box. The Properties window displays different types of editing fields, depending on the needs of a particular property.



txtNum1 System.Windows.Forms.TextBox

⊞ (ApplicationSettings)	
⊞ (DataBindings)	
(Name)	**txtNum1**
AcceptsReturn	False
AcceptsTab	False
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	Top, Left
AutoCompleteCustomSource	**(Collection)**
AutoCompleteMode	None
AutoCompleteSource	None
BackColor	☐ Window
BorderStyle	Fixed3D
CausesValidation	True
CharacterCasing	Normal
ContextMenuStrip	(none)
Cursor	IBeam
Dock	None
Enabled	True
⊞ Font	Microsoft Sans Serif, 8.25pt
ForeColor	■ WindowText
GenerateMember	True
HideSelection	True
ImeMode	NoControl
⊞ Lines	**String[] Array**
⊞ Location	**135, 43**
Locked	False
⊞ Margin	3, 3, 3, 3
⊞ MaximumSize	0, 0
MaxLength	32767
⊞ MinimumSize	0, 0
Modifiers	Private
Multiline	False
PasswordChar	
ReadOnly	False
RightToLeft	No
ScrollBars	None

Text
The text associated with the control.

Setting the properties – Step 4

- Select the first textbox and set the name as txtNum1 on the property window.
- Similarly set the names of other two textboxes as txtNum2 and txtResult respectively.
- For Labels, Set the text property to Number 1, Number 2 and Result respectively
- For Buttons, set the properties as shown in the given table.

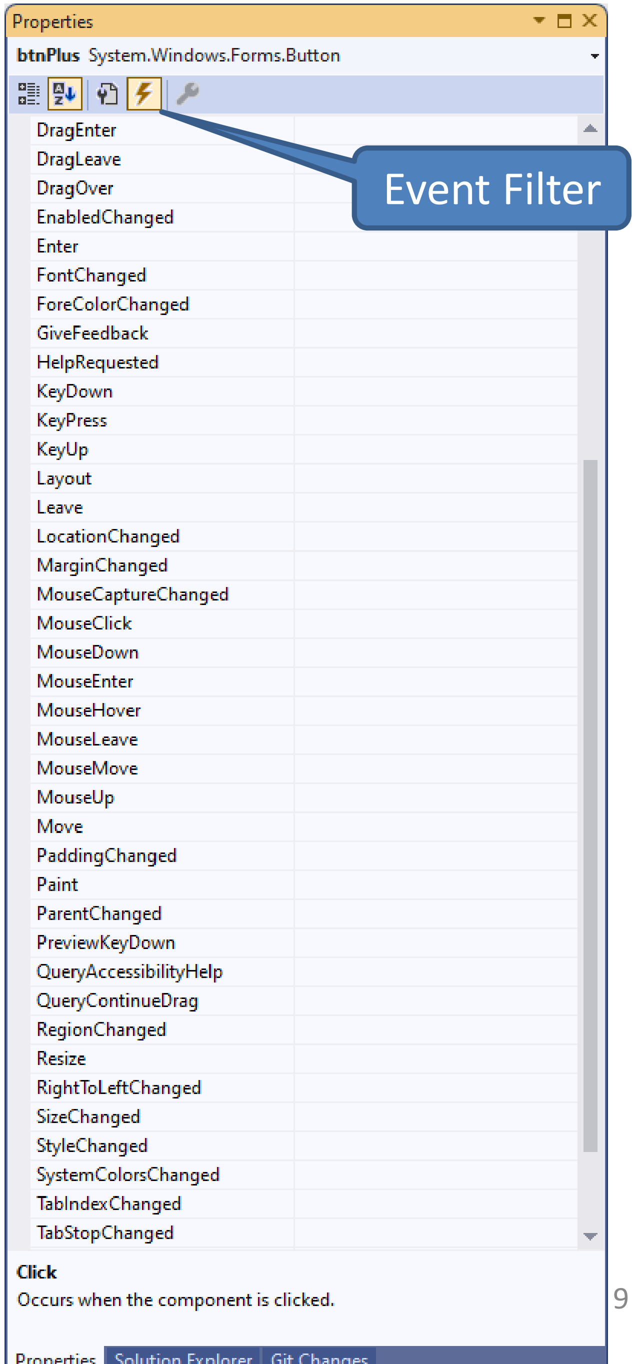
Name	Text
btnPlus	+
btnMinus	-
btnMul	X
btnDiv	/

Events

- An event is a signal that informs an application that something important has occurred.
- For example, when a user clicks a control on a form, the form can raise a Click event and call a procedure that handles the event. Events also allow separate tasks to communicate.
- Eg: click, double click, Key Press etc..

List of Events of a Button

- You can filter List of events on property window by clicking the event filter

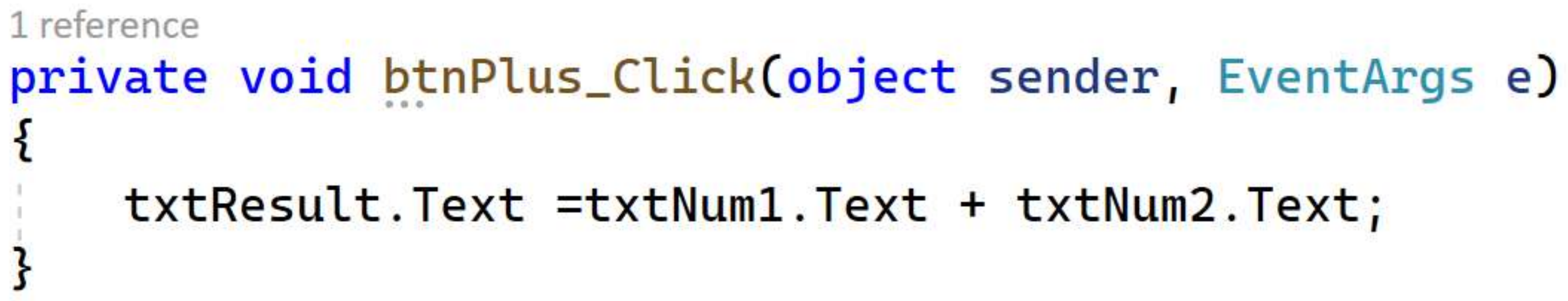

Properties
btnPlus System.Windows.Forms.Button

DragEnter
DragLeave
DragOver
EnabledChanged
Enter
FontChanged
ForeColorChanged
GiveFeedback
HelpRequested
KeyDown
KeyPress
KeyUp
Layout
Leave
LocationChanged
MarginChanged
MouseCaptureChanged
MouseClick
MouseDown
MouseEnter
MouseHover
MouseLeave
MouseMove
MouseUp
Move
PaddingChanged
Paint
ParentChanged
PreviewKeyDown
QueryAccessibilityHelp
QueryContinueDrag
RegionChanged
Resize
RightToLeftChanged
SizeChanged
StyleChanged
SystemColorsChanged
TabIndexChanged
TabStopChanged

Click
Occurs when the component is clicked.

Adding Click events – Step 5

- Double click on plus button to open code editor for click event and type the following code.



```
1 reference
private void btnPlus_Click(object sender, EventArgs e)
{
    txtResult.Text =txtNum1.Text + txtNum2.Text;
}
```

Running the application

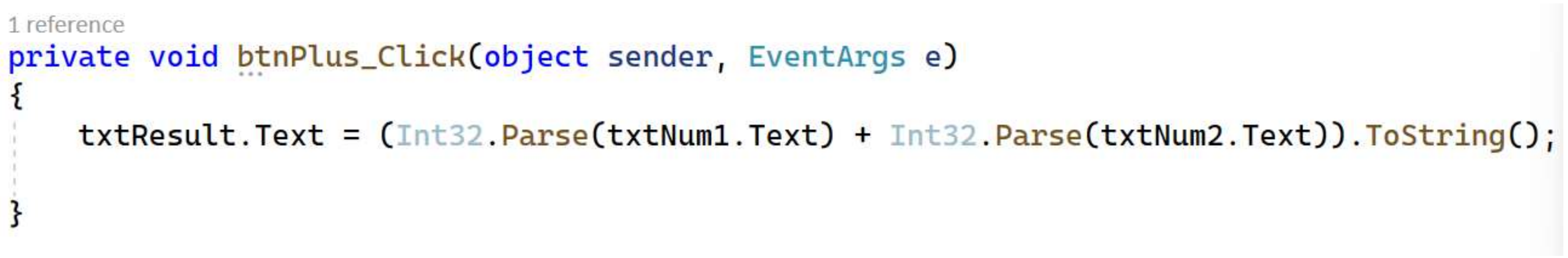
- Input values for Number1 and Number 2 and check the Result by clicking the plus button.
- For eg:

Number 1	Number 2	Result
Com	puter	Computer
12	6	126
I am	Kumar	I amKumar

Addition Vs Concatenation

- In the above example ‘+’ operator works as a string Concatenation operator, not an addition operator. Because both operands are string.
- If you need addition operation, you have to convert both operands to numeric values such as integer, double etc..

- Change the code of the click event as shown below to do arithmetic operations like
- +, -, * and /



```
1 reference
private void btnPlus_Click(object sender, EventArgs e)
{
    txtResult.Text = (Int32.Parse(txtNum1.Text) + Int32.Parse(txtNum2.Text)).ToString();

}
```

- Run the application and input integer values to Number 1 and Number 2

Int32.Parse Method

- Converts the string representation of a number to its 32-bit signed integer equivalent.

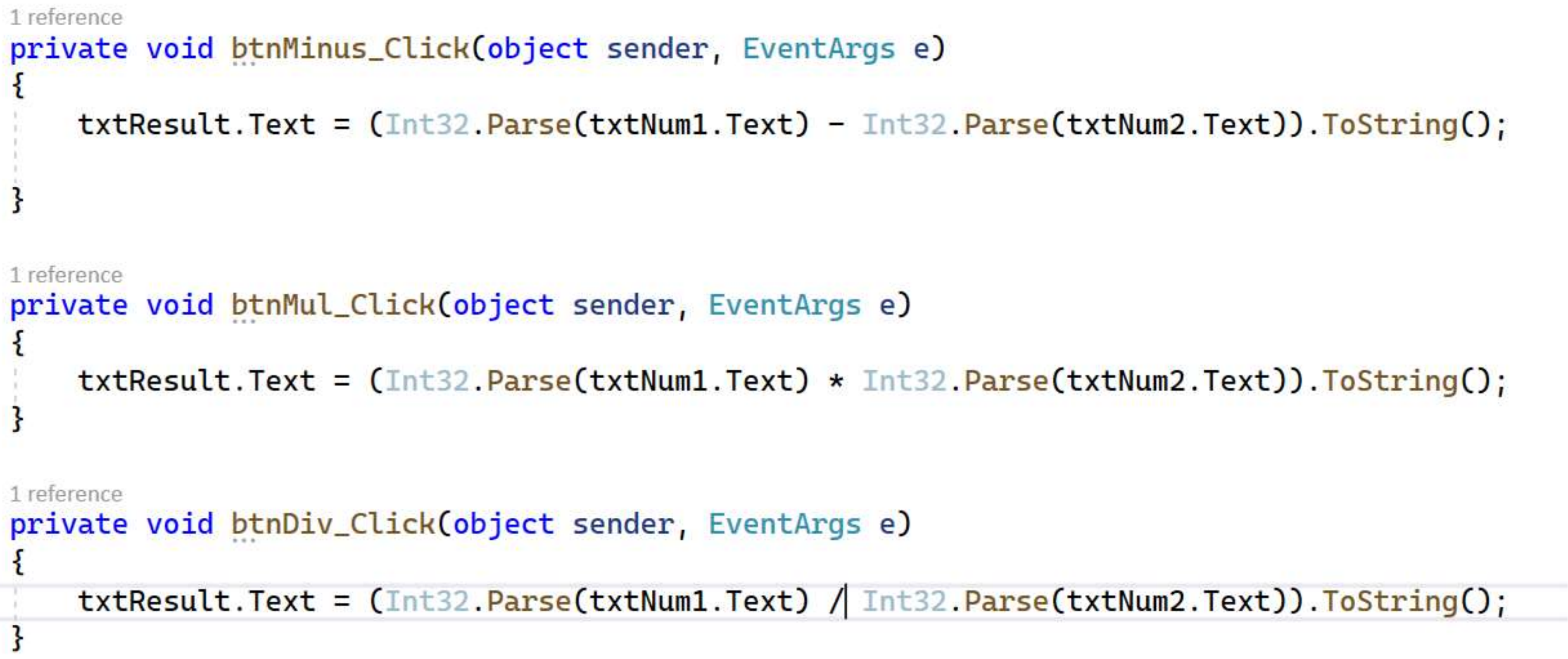
toString Method

- It converts an object to its string representation so that it is suitable for display.

C# Data types

C# type keyword	.NET type
bool	System.Boolean
byte	System.Byte
sbyte	System.SByte
char	System.Char
decimal	System.Decimal
double	System.Double
float	System.Single
int	System.Int32
uint	System.UInt32
nint	System.IntPtr
nuint	System.UIntPtr
long	System.Int64
ulong	System.UInt64
short	System.Int16
string	System.String

- Add click events to other buttons and type the code as given below:



```
1 reference
private void btnMinus_Click(object sender, EventArgs e)
{
    txtResult.Text = (Int32.Parse(txtNum1.Text) - Int32.Parse(txtNum2.Text)).ToString();

}

1 reference
private void btnMul_Click(object sender, EventArgs e)
{
    txtResult.Text = (Int32.Parse(txtNum1.Text) * Int32.Parse(txtNum2.Text)).ToString();
}

1 reference
private void btnDiv_Click(object sender, EventArgs e)
{
    txtResult.Text = (Int32.Parse(txtNum1.Text) / Int32.Parse(txtNum2.Text)).ToString();
}
```

- / operator performs an integer division if both operands are integers.
- Eg:
    - ❖3/4 = 0
    - ❖4/4 =1
    - ❖5/4=1

C# Arithmetic Operators

Operator	Name	Description	Example
+	Addition	Adds together two values	x + y
-	Subtraction	Subtracts one value from another	x - y
*	Multiplication	Multiplies two values	x * y
/	Division	Divides one value by another	x / y
%	Modulus	Returns the division remainder	x % y
++	Increment	Increases the value of a variable by 1	x++
--	Decrement	Decreases the value of a variable by 1	x--

C# assignment Operators

Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
&=	x &= 3	x = x & 3
|=	x |= 3	x = x | 3
^=	x ^= 3	x = x ^ 3
>>=	x >>= 3	x = x >> 3
<<=	x <<= 3	x = x << 3

C# Comparison Operators

Operator	Name	Example
==	Equal to	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

C# Logical Operators

Operator	Name	Description	Example
&&	Logical and	Returns true if both statements are true	x < 5 && x < 10
||	Logical or	Returns true if one of the statements is true	x < 5 || x < 4
!	Logical not	Reverse the result, returns false if the result is true	!(x < 5 && x < 10)

Bitwise Operators

Bitwise operator works on bits and perform bit by bit operation. The truth tables for
& - AND,
| - OR, and
^ - XOR are as follows −

p	q	p & q	p | q	p ^ q
0	0	0	0	0
0	1	0	1	1
1	1	1	1	0
1	0	0	1	1

Bitwise Operators

Assume if A = 60; and B = 13; then in the binary format they are as follows
A = 0011 1100 B = 0000 1101

Operator	Description	Example
&	Binary AND Operator copies a bit to the result if it exists in both operands.	(A & B) = 12, which is 0000 1100
|	Binary OR Operator copies a bit if it exists in either operand.	(A | B) = 61, which is 0011 1101
^	Binary XOR Operator copies the bit if it is set in one operand but not both.	(A ^ B) = 49, which is 0011 0001
~	Binary Ones Complement Operator is unary and has the effect of 'flipping' bits.	(~A) = -61, which is 1100 0011 in 2's complement due to a signed binary number.
<<	Binary Left Shift Operator. The left operands value is moved left by the number of bits specified by the right operand.	A << 2 = 240, which is 1111 0000
>>	Binary Right Shift Operator. The left operands value is moved right by the number of bits specified by the right operand.	A >> 2 = 15, which is 0000 1111

Operator Precedence in C#

Category	Operator	Associativity
Postfix	() [] -> . ++ - -	Left to right
Unary	+ - ! ~ ++ - - (type)* & sizeof	Right to left
Multiplicative	* / %	Left to right
Additive	+ -	Left to right
Shift	<< >>	Left to right
Relational	< <= > >=	Left to right
Equality	== !=	Left to right
Bitwise AND	&	Left to right
Bitwise XOR	^	Left to right
Bitwise OR	|	Left to right
Logical AND	&&	Left to right
Logical OR	||	Left to right
Conditional	?:	Right to left
Assignment	= += -= *= /= %=>>= <<= &= ^= |=	Right to left
Comma	,	Left to right

Thank You