Republic of the Philippines
CEBU TECHNOLOGICAL UNIVERSITY
MAIN CAMPUS
M.J. Cuenco Ave. Cor. R. Palma Street, Cebu City, Philippines
Website: http://www.ctu.edu.ph

COLLEGE OF COMPUTER, INFORMATION AND COMMUNICATIONS TECHNOLOGY

Microsoft

# ASP.NET

## Module 4

## Operators

## I. Preparations

At the end of this module students will:

- construct arithmetic, relational and logical expression;
- construct simple condition and compound condition

## II. Presentation

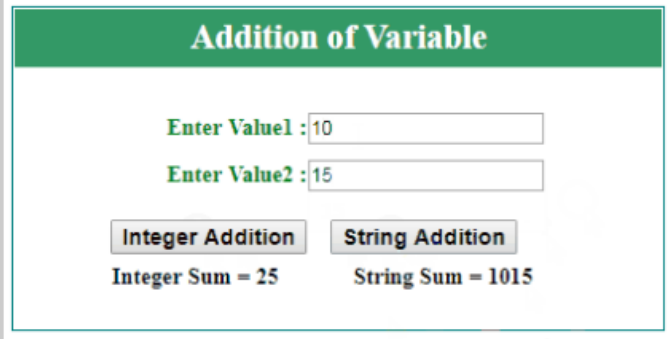
Operator are used to perform mathematical or logical operation on variable. Operator is a **symbol that perform specific mathematical or logical operation** on variable in any programming language.

**List of C# Operators**

- Arithmetic Operators
- Logical Operators
- Relational or Comparison Operators
- Assignment Operators

| Operator | Description |
|---|---|
| * | Multiplication |
| / | Division |
| % | Reminder |
| + | Addition |
| - | Subtraction |

**Example #1 – Addition**



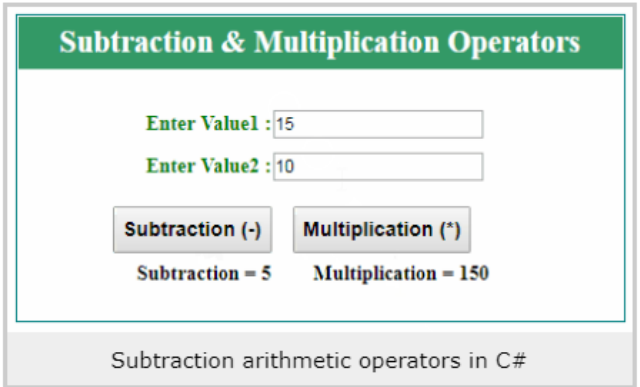

Addition of Variable Example.

Above example as show screen, there are two values 10 and 15 to the textbox and on **Integer Addition** and **String Addition** buttons. The results, for **Integer Addition** button is **15** while **String Addition** button is **1015.**

Here is the c# code for above addition example:

```
protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void btnIntegeraddition_Click(object sender, EventArgs e)
    {
        int firstno;
        int secondno;

        firstno = Convert.ToInt32(TextBox1.Text);
        secondno = Convert.ToInt32(TextBox2.Text);

        int addanswer = firstno + secondno;

        lblintegeranswer.Text ="Integer Sum = "+ addanswer.ToString();
    }

    protected void btnStringaddition_Click(object sender, EventArgs e)
    {
        string firstno;
        string secondno;

        firstno = TextBox1.Text;
        secondno = TextBox2.Text;

        lblstringanswer.Text ="String Sum = " +  firstno + secondno;
    }
```

**Example #2 – Subtraction and Multiplication**

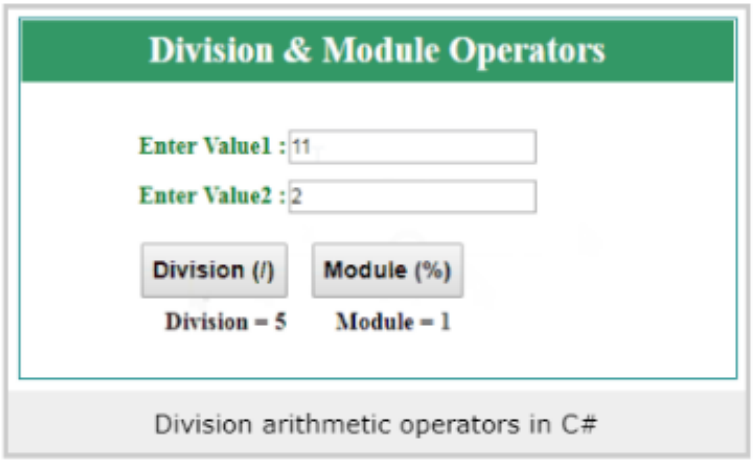



Subtraction arithmetic operators in C#

Here is the code for above example:

```
protected void btnsubtraction_Click(object sender, EventArgs e)
    {
        int firstno;
        int secondno;

        firstno = Convert.ToInt32(TextBox1.Text);
        secondno = Convert.ToInt32(TextBox2.Text);

        int answer = firstno - secondno;

        lblsub.Text = "Subtraction = " + answer.ToString();

    }
    protected void btnmultiplication_Click(object sender, EventArgs e)
    {
        int firstno;
        int secondno;

        firstno = Convert.ToInt32(TextBox1.Text);
        secondno = Convert.ToInt32(TextBox2.Text);

        int answer = firstno * secondno;

        lblmul.Text = "Multiplication = " + answer.ToString();
    }
```

**Example #3 – Division and Module**




Division arithmetic operators in C#

C# code for example

```
protected void btndivision_Click(object sender, EventArgs e)
    {
        int firstno;
        int secondno;

        firstno = Convert.ToInt32(TextBox1.Text);
        secondno = Convert.ToInt32(TextBox2.Text);

        int answer = firstno / secondno;

        lbldiv.Text = "Division = " + answer.ToString();

    }
    protected void btnmodule_Click(object sender, EventArgs e)
    {
        int firstno;
        int secondno;

        firstno = Convert.ToInt32(TextBox1.Text);
        secondno = Convert.ToInt32(TextBox2.Text);

        int answer = firstno % secondno;

        lblmod.Text = "Module = " + answer.ToString();

    }
```

**Logical Operators**

The C# language supports basic three logical operators. The logical operators returns true or false as output.

The logical operators are used when we have conditional statement such as ***if statement***.

**C# Supports following logical operators**

Assume variable X is true and variable Y is false then

| Operator | Meaning | Example |
|---|---|---|
| && | Logical AND | if(X && Y) then false |
| \|\| | Logical OR | if(X \|\| Y) then true |
| ! | Logical NOT | if !(X && Y) then true |

## The *Convert* class

The Convert class includes different methods which convert base data type to another base data type.

The Convert class includes the following methods to convert from different data types to int type.

**Convert.ToInt16()**
**Convert.ToInt32()**
**Convert.ToInt64()**

The Convert.ToInt16() method returns the 16-bit integer e.g. short, the Convert.ToInt32() returns 32-bit integers e.g. int and the Convert.ToInt64() returns the 64-bit integer e.g. long.

Example: Convert string to int using Convert class

```
Convert.ToInt16("100"); // returns short
Convert.ToInt16(null);//returns 0

Convert.ToInt32("233300");// returns int
Convert.ToInt32("1234",16); // returns 4660 - Hexadecimal of 1234

Convert.ToInt64("1003232131321321");//returns long

// the following throw exceptions
Convert.ToInt16("");//throws FormatException
Convert.ToInt32("30,000"); //throws FormatException
Convert.ToInt16("(100)");//throws FormatException
Convert.ToInt16("100a"); //throws FormatException
Convert.ToInt16(2147483649);//throws OverflowException
```

Pros:

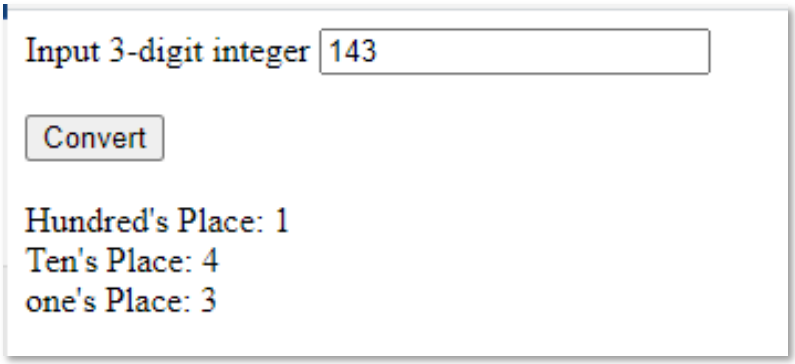
Converts from any data type to integer.
Converts null to 0, so not throwing an exception.

Cons:

Input string must be valid number string, cannot include different numeric formats. Only works with valid integer string.
Input string must be within the range of called IntXX method e.g. Int16, Int32, Int64.
The input string cannot include parenthesis, comma, etc.
Must use a different method for different integer ranges e.g. cannot use the Convert.ToInt16() for the integer string higher than "32767".
Visit Convert class for more information.

## III. Practice. Problem Solving

Construct Webform that will accept 3-digit integer and will display the one's, tent's and the hundred's place.




## IV. **Performance.** Problem Solving

Construct Webform that will accept the price of an item and the amount tendered. Display the change and its coin denomination.

Example: