C#-1

C# Language Fundamentals

Objectives

- How to write a simple program using C#?
- Features of Main Function
- How to pass Command Line Arguments
- How to work on Multiple Mains of a Program.
- What is Namespaces
- Different types of variables and constants
- Control flow statements
- Enumerations
- Arrays and different types of arrays
- Console I/O
- Programming guidelines of C#.

What's C#?

- C# (pronounced as See-Sharp) is a language that has been developed for building a wide range of enterprise applications that run on the .NET Framework.
- Simply it is a language for building .NET based Applications.
- Combines features of multiple languages and more:
  - RAD of VB
  - Object Oriented Feature of C++.
  - Elegance of Java.
- Above all, supports Component Architecture.
- It is a type-safe, case-sensitive language that supports all Object Oriented Programming features

Features of the C# Language

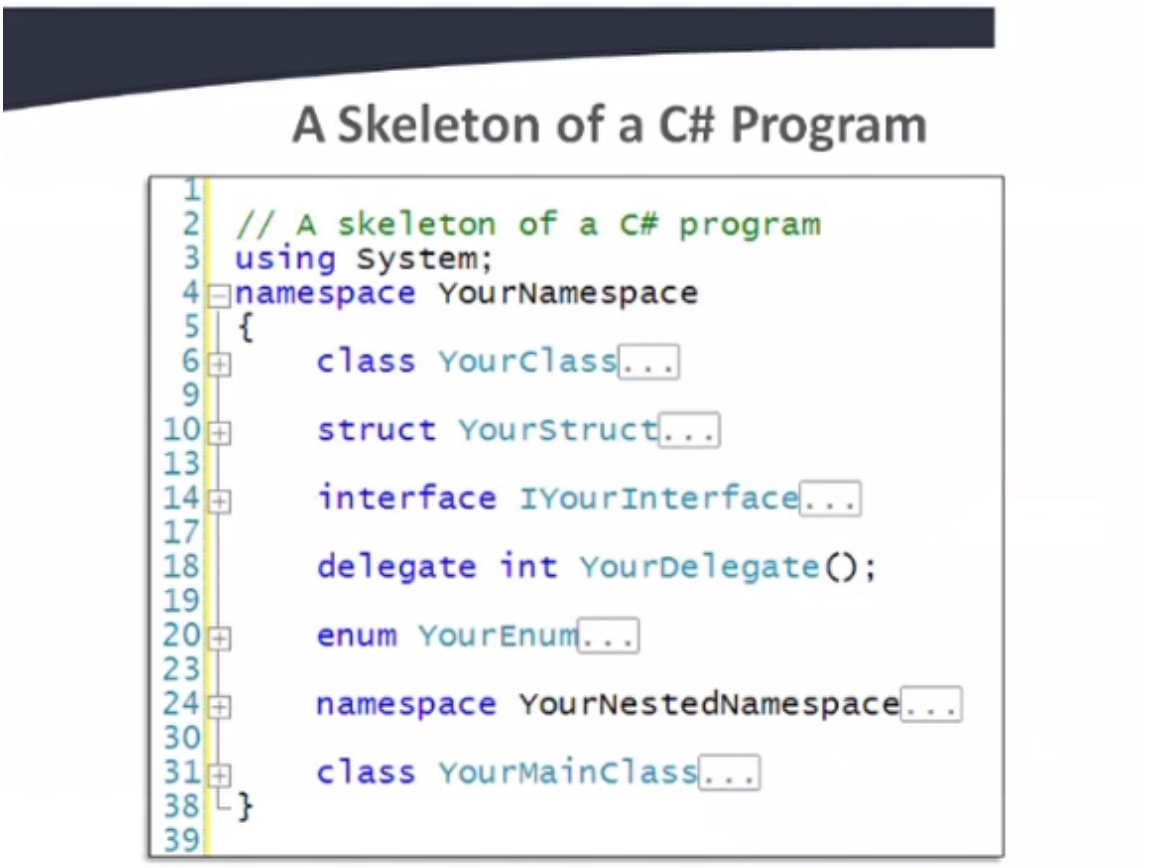
- Pure Object Oriented Programming language.
- Component Architecture.
- Strict Type checking and no implicit casts.
- No uninitialized variables and Safe Bound Checks.
- You can overload operators of the language.
- Template mechanisms through Generics.
- Event based programming using delegates.
- Overloads [] operator with Indexers.

Architectural History of C#

- Anders Hejlsberg.
  - C#'s principal designer and lead architect at Microsoft.
  - He designed
    - Visual J++
    - Borland Delphi
    - Turbo Pascal
    - .NET CLR (Core)
    - C# Language

Anders Hejlsberg, Microsoft Technical Fellow and Chief Architect, C#

This is a super Man- just read about this guy you will love him



in a particular C# program we have a using statement on the top

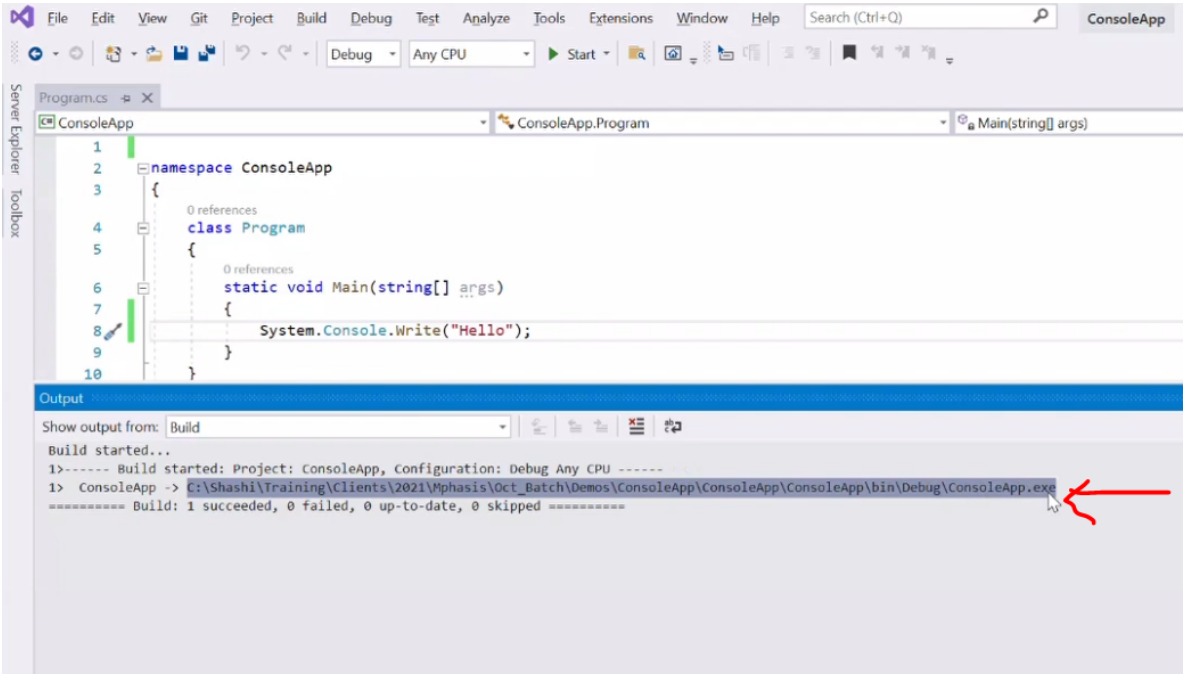
What is a need of Using statements ?

suppose i want to print any thing on the screen, so to print anything on the screen we will be using a **System** Library which is a part of .net core library
so, system library has a system.console.write
console is a class and this class has a method called Write or WriteLine
What is a Console Application?
--Console is a black and white application like command prompt, this is called console screen
and any application which you developed using console application, will gone to use that console screen to read or display the information

```
//using System;

namespace ConsoleApp
{
    0 references
    class Program
    {
        0 references
        static void Main()
        {
            Console.WriteLine("Hi\n\tShashi\n\t\tHow\n\t\t\tAre\n\t\t\t\tYou?");
            Console.WriteLine("Hi");
            Console.WriteLine("Hello");
            Console.Write("Welcome");
            Console.ReadLine();
        }
    }
}
```

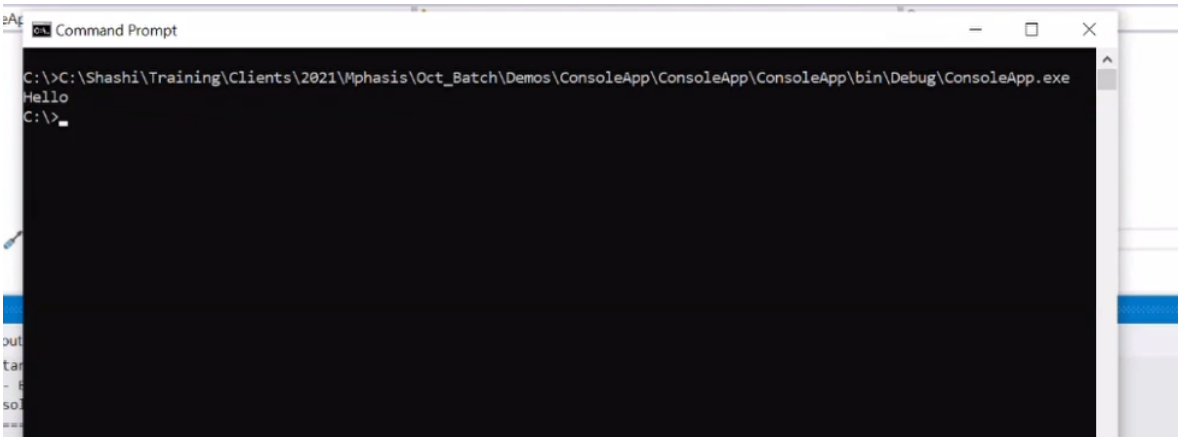
To Run this program we press**ctrl + f5 -----it compiles it builds and then it executes , now which is the file it is executing**



it is executing this file can you see consoleapp.exe

now can i execute it directly --yes

open command prompt ------>go to c drive for this type **cd/** ---------->then paste this entire path and press enter



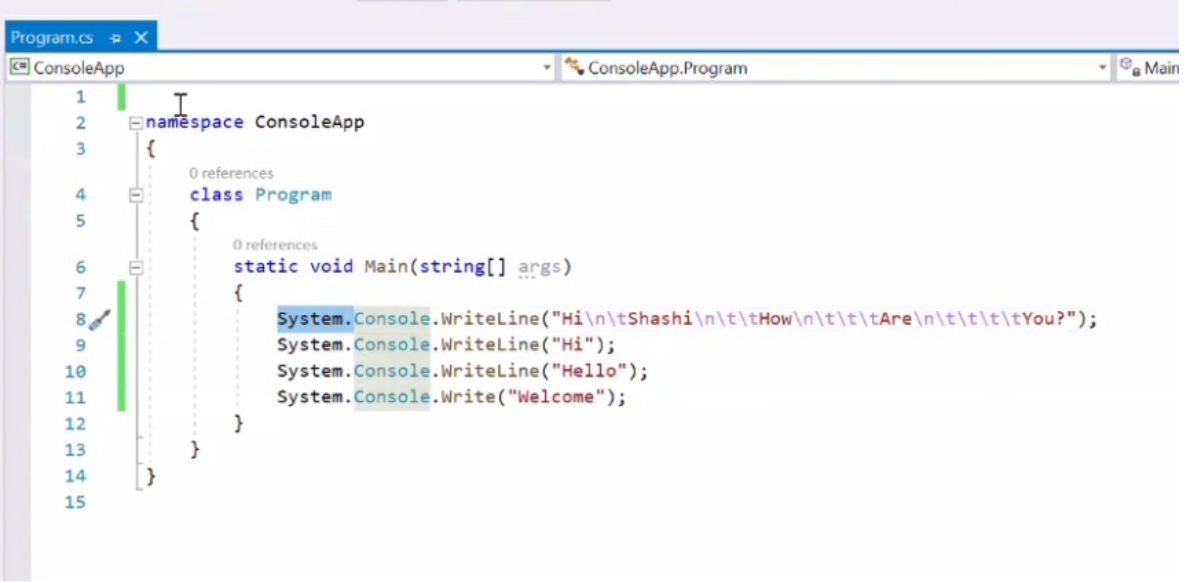
write--------->write function will print the content in same line ,to move content to the next line you can use **\n ----**it is called escape sequence, it will change the sequence of printing

similarly **\t** it is for tab space

WriteLine----->instead of using \n here we can use writeline function

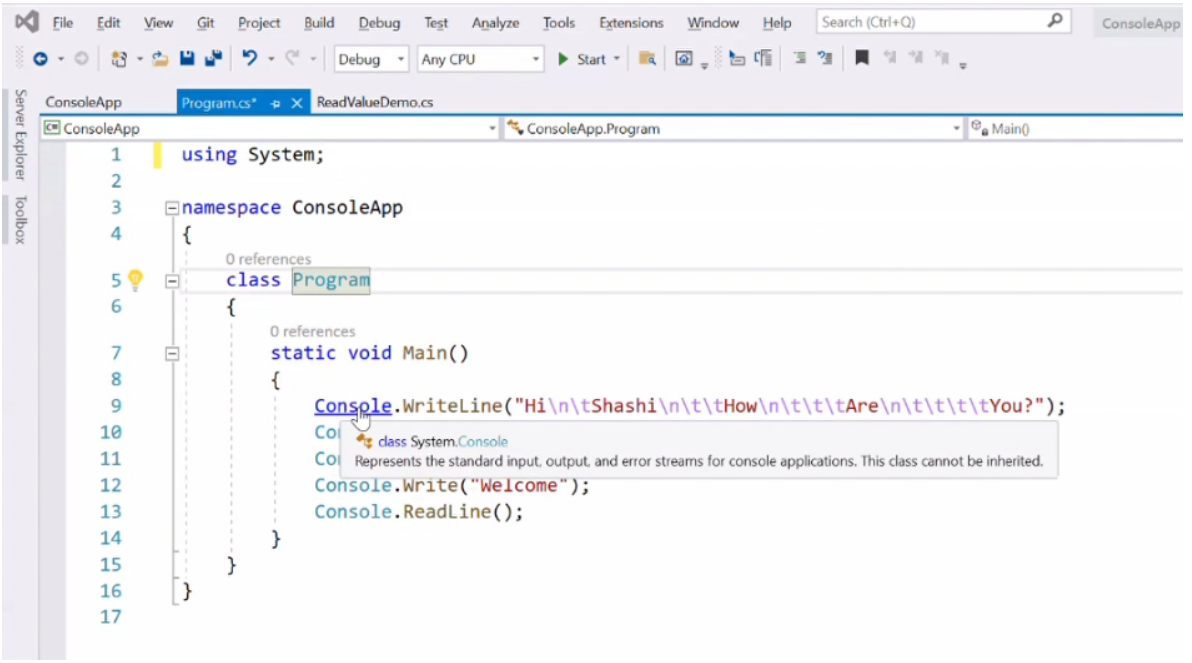
now to see the power of ide just write -------WriteL and hold ctrl key and press spacebar automatically you will get writeLine

so to get intellisense we use ctrl+space



you can see many times i have write system. system. .....here ,so instead of this i can use using system

so if i write using system on the top then no need of me to write system again



What is System ?

So system is a namespace , namespace is compared to packages in java but there is lot of difference between java package and namespace.

What is the main difference?

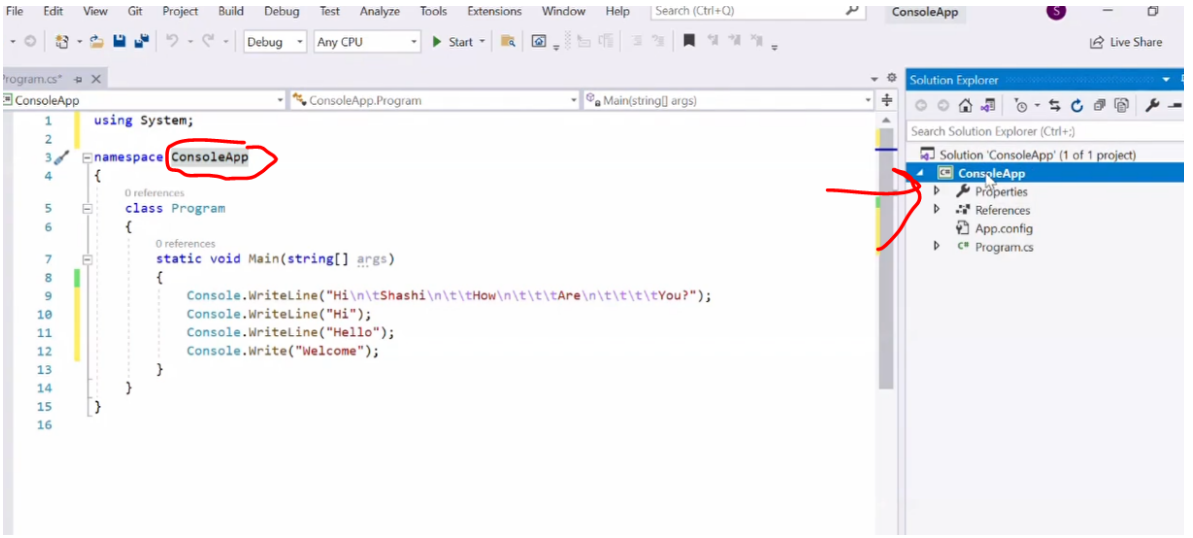
so packages in java are physical folders , we need to have physical folder created , but whereas it is not the case, so you can create your own namespaces and they are not at all physical locations it is just a logical location

What is the use of this things ?

to bind all the logically related classes together in one namespace, its like a bucket its like a container in that container i can put all the related classes

How do you create a namespace?

just say namespace followed by namespace name{.....}

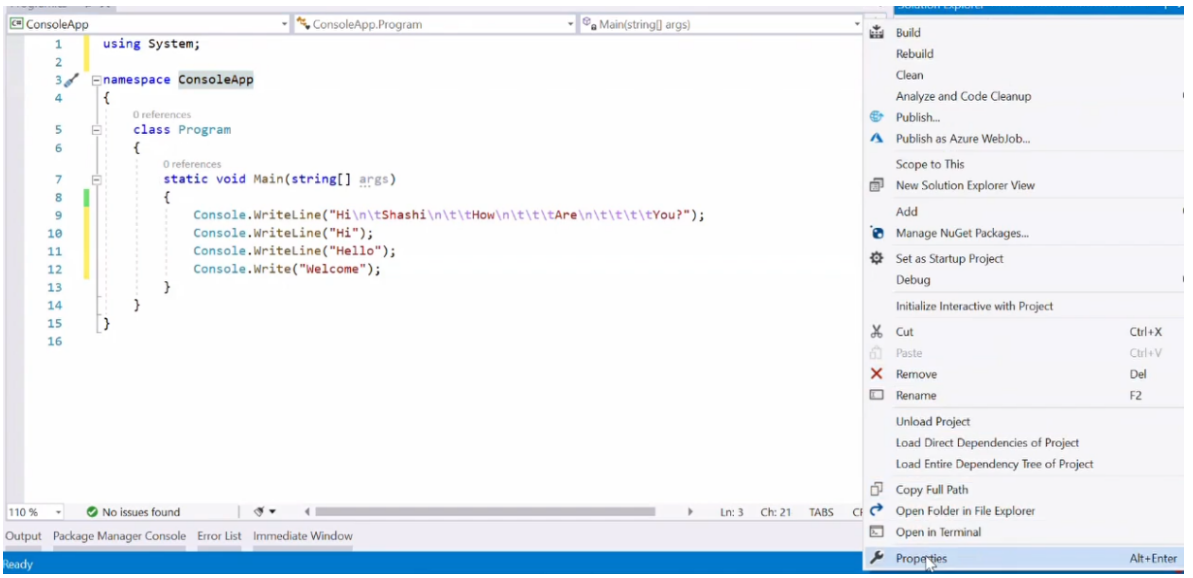


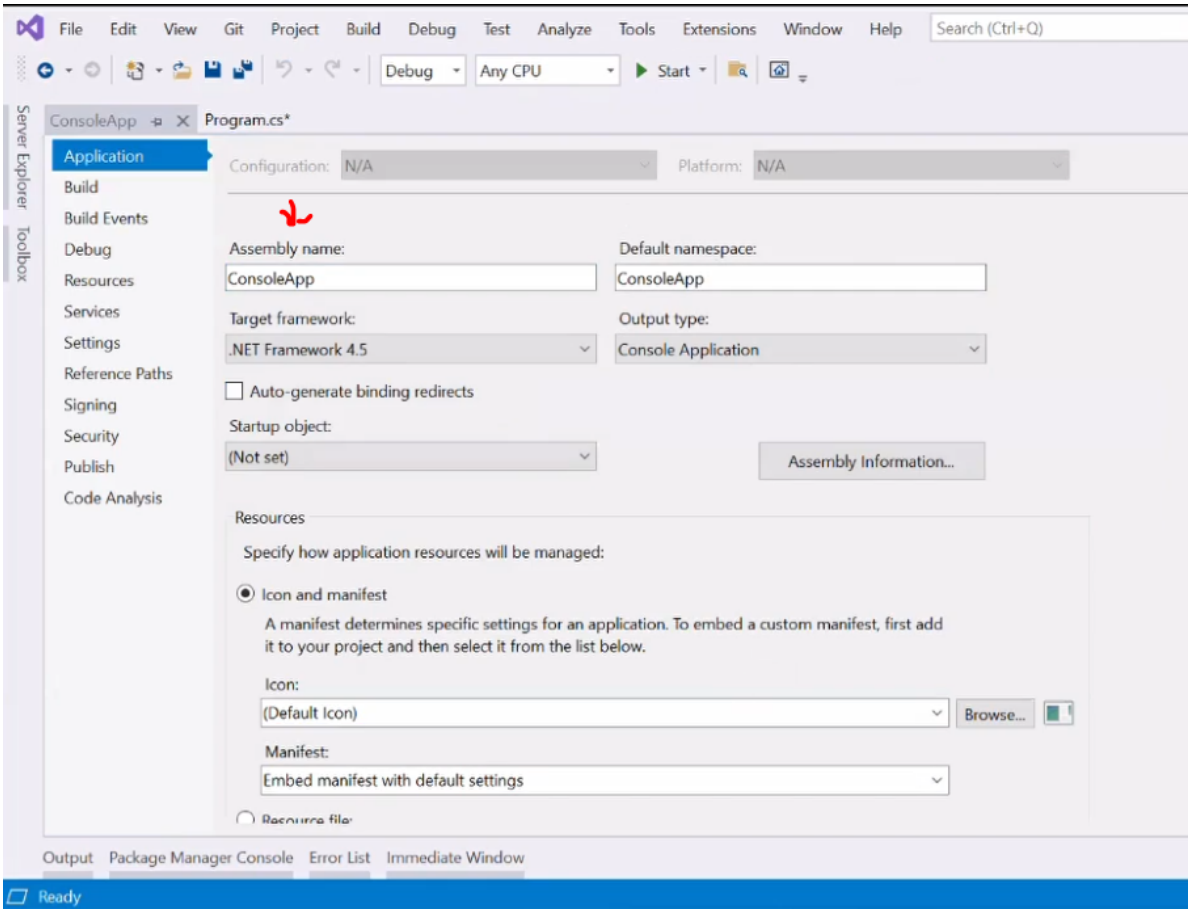
This namespace is the name of my project

1 solution can have many projects and each project may or may not have different namespaces

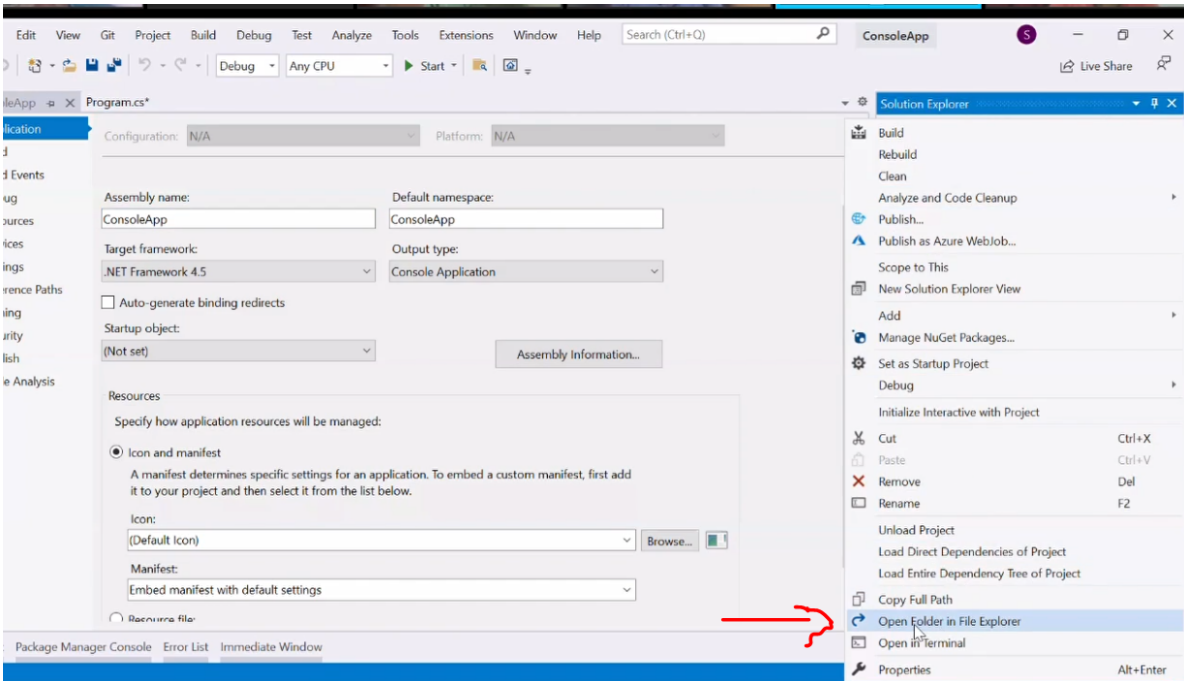
so, usually namespace name will be the assembly name whatever we are giving here

just right click here and go to properties

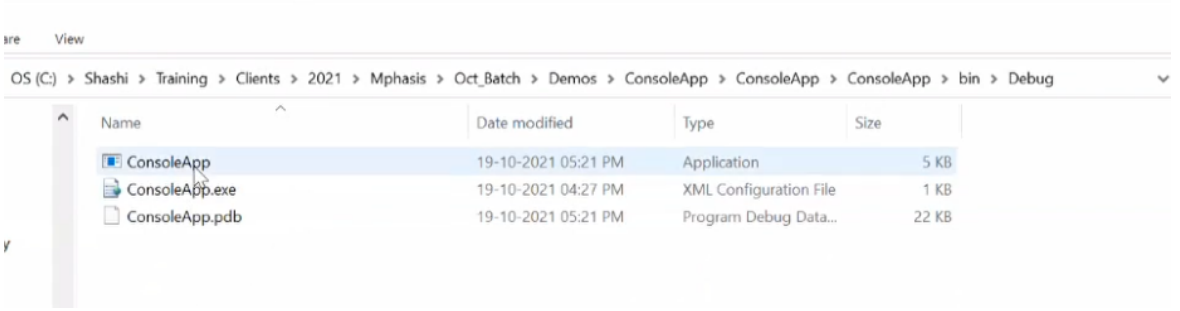




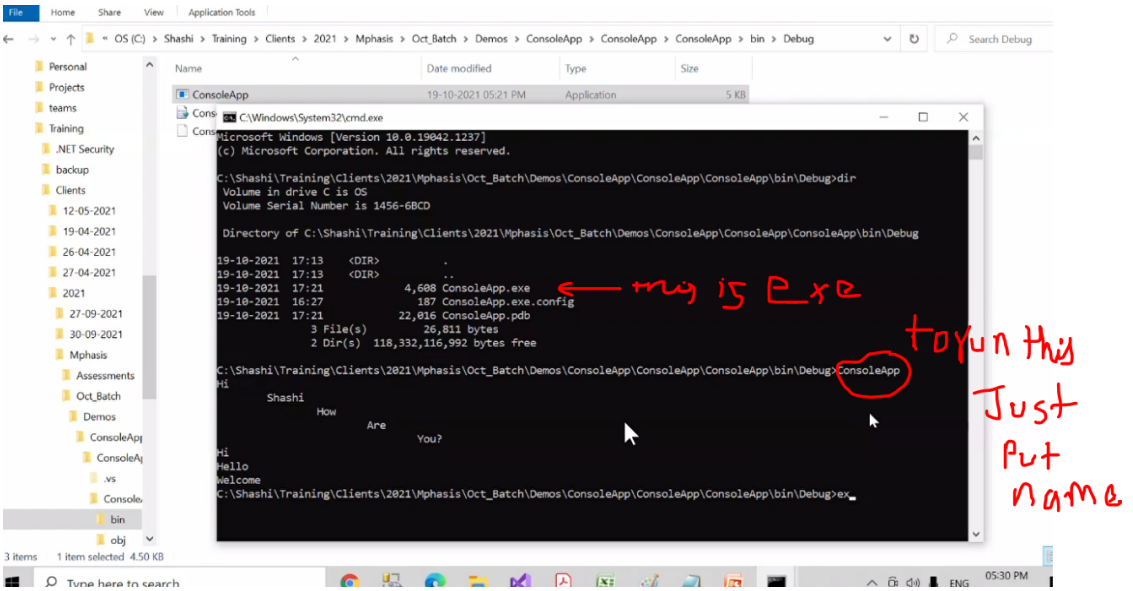
To know the location of this project right click and click on open file location



now assembly file is stored in side---->bin------>debug



in assembly we have 2 types of assembly

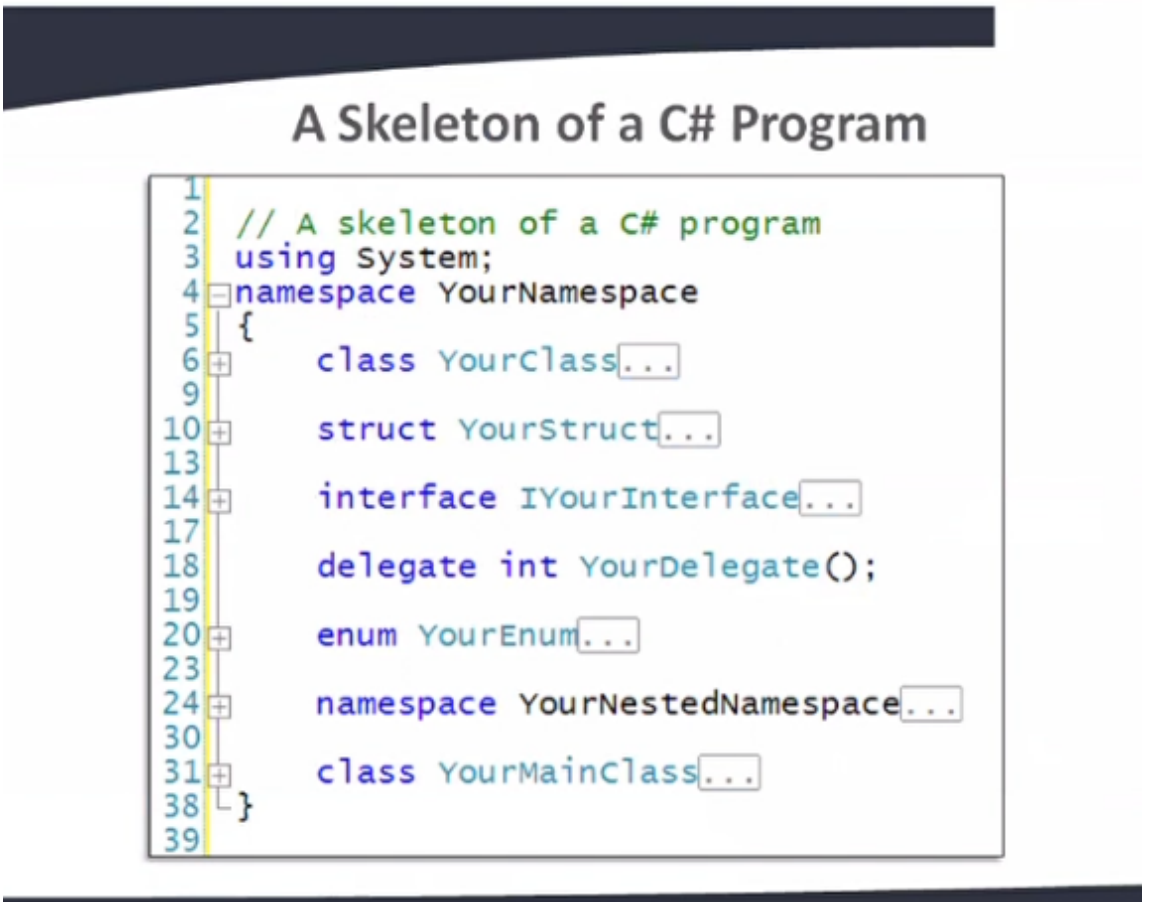
1. **.EXE**
2. **.DLL ------is Dynamic Link Library**
3. 

you can also double click on this .exe to open

If Some Body ask you to give this file so you can copy this 3 files and you can give it to your end user

now it will just open and close, so to keep it open till user press any key we write Read or ReadLine(), you can compare this with scanner class in java

```
1   using System;
2
3   namespace ConsoleApp
4   {
        0 references
5       class Program
6       {
            0 references
7           static void Main()
8           {
9               Console.WriteLine("Hi\n\tShashi\n\t\tHow\n\t\t\tAre\n\t\t\t\tYou?");
10              Console.WriteLine("Hi");
11              Console.WriteLine("Hello");
12              Console.Write("Welcome");
13              Console.ReadLine();
14          }
15      }
16  }
17
```



SO this is about using

Why are we using ,USING?

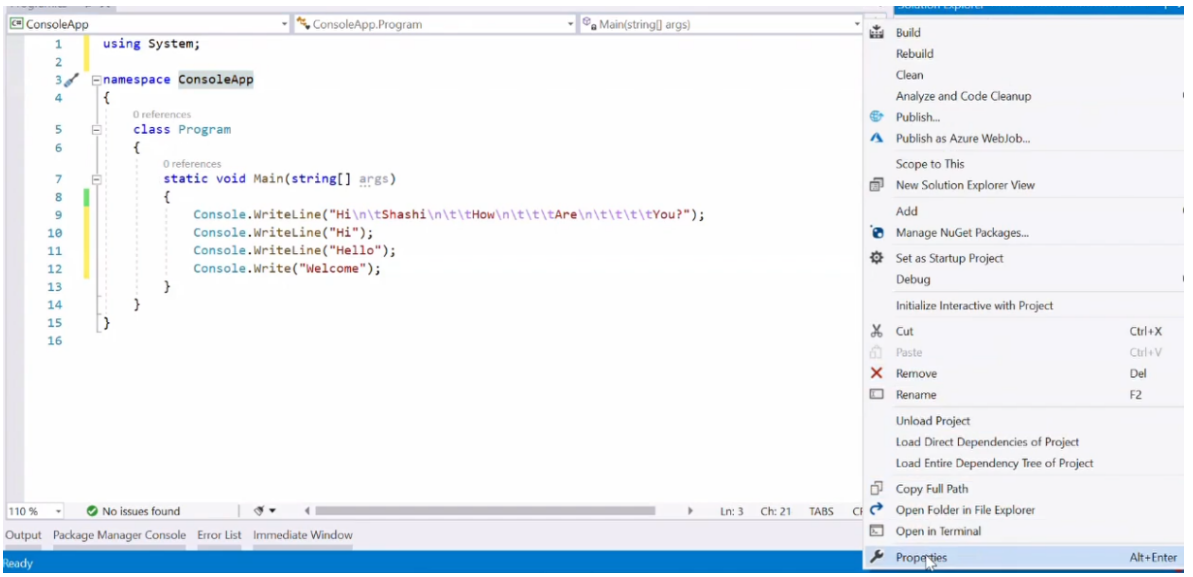
whenever we have a particular package so instead of rewriting the package again and again we always say

using followed by package name
NameSpace----is to bind the logically related classes together
Class Followed by className will help me to define my classes

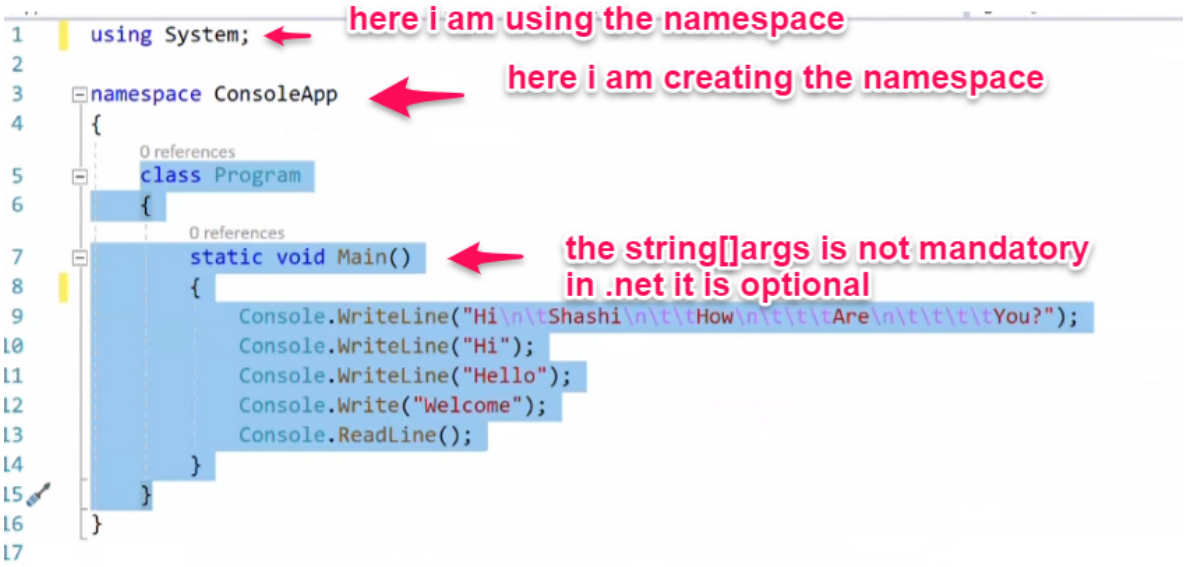
Remember in your C# you cannot create a variables so anything and every thing should be the part of your class, if it is not a class then it can be struct or can be a interface or a deligate or enum or namespace or class so only these things can be there

you cannot say int a , int b ....not in java not in c#----------anything and everything should be the part of class

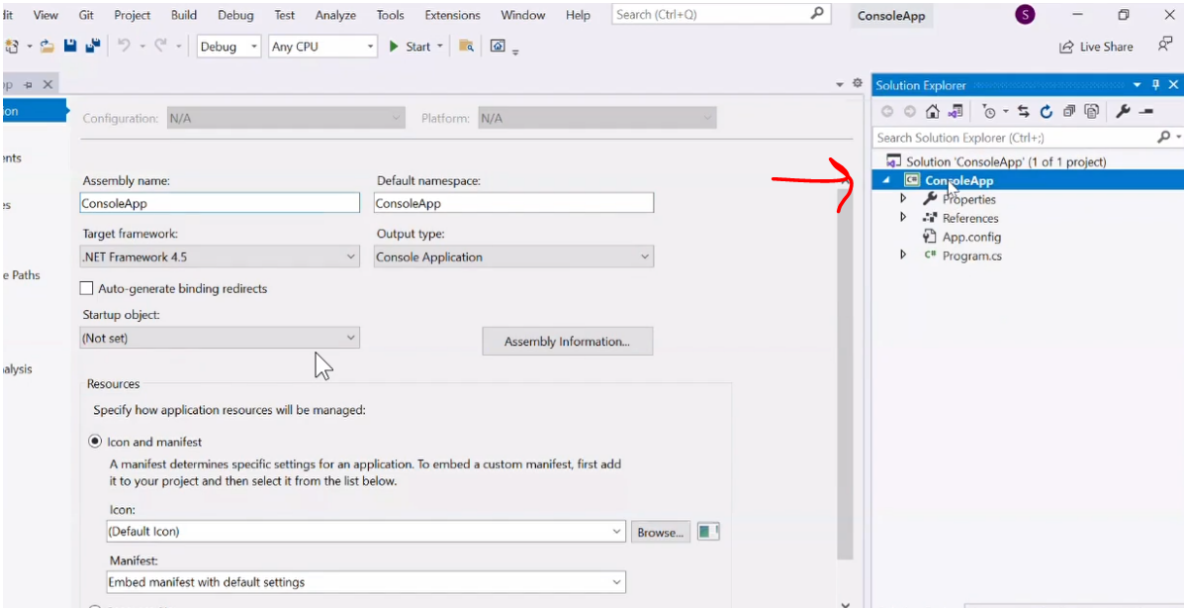
so outside the class you can have only your namespaces or enum or structure or interface or deligates.....nothing else



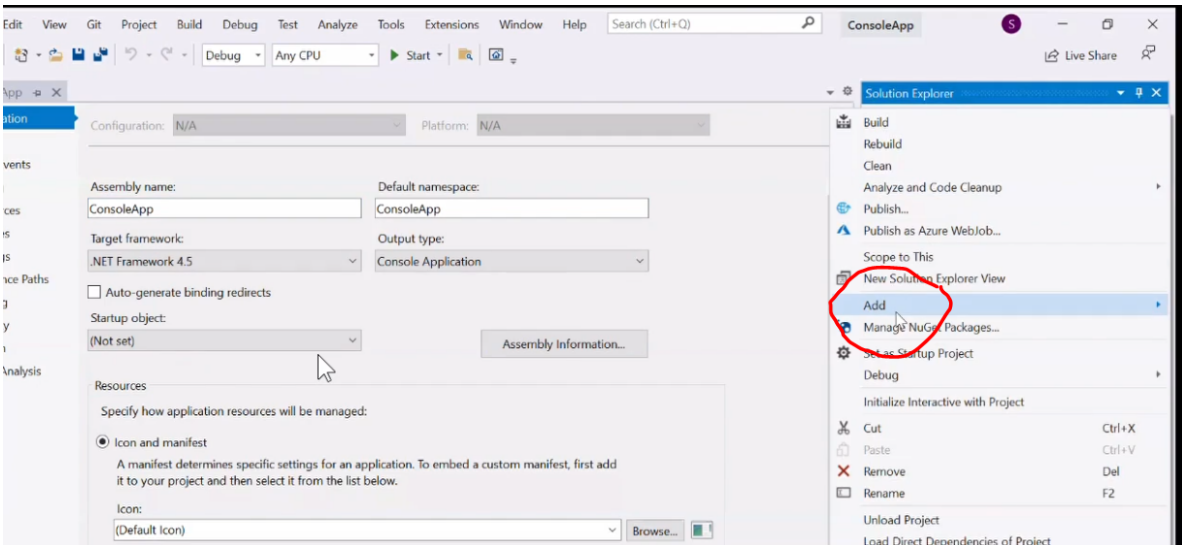
here you can see that we are saying **static void Main(string[]**args) this becomes the entry point to my entire application



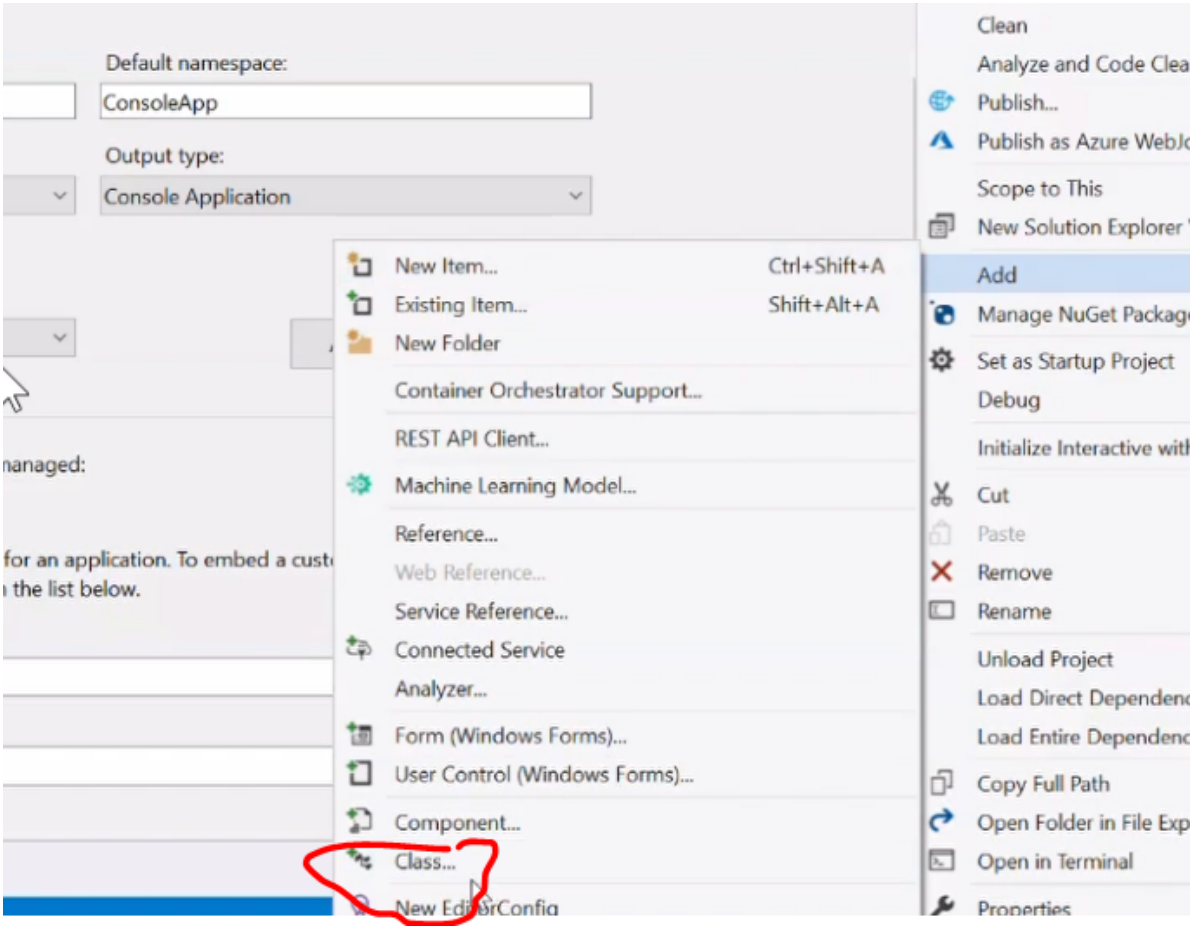
now to create new class ,right click here

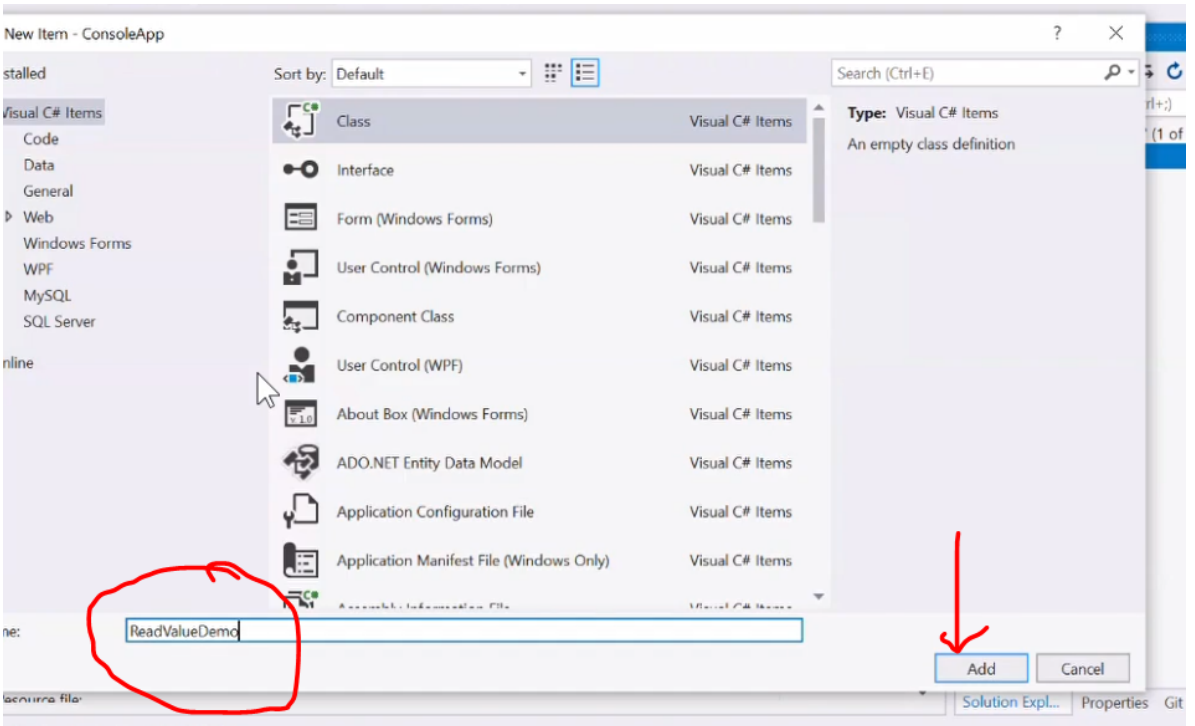


then click on add



then class


Default namespace:
ConsoleApp
Output type:
Console Application
New Item... Ctrl+Shift+A
Existing Item... Shift+Alt+A
New Folder
Container Orchestrator Support...
REST API Client...
Machine Learning Model...
Reference...
Web Reference...
Service Reference...
Connected Service
Analyzer...
Form (Windows Forms)...
User Control (Windows Forms)...
Component...
Class...
Clean
Publish...
Scope to This
Add
Manage NuGet Packag
Set as Startup Project
Debug
Cut
Paste
Remove
Rename
Unload Project
Copy Full Path
Open Folder in File Exp
Open in Terminal
Properties


New Item - ConsoleApp
Sort by: Default
Search (Ctrl+E)
Visual C# Items
Code
Data
General
Web
Windows Forms
WPF
MySQL
SQL Server
Class Visual C# Items
Interface Visual C# Items
Form (Windows Forms) Visual C# Items
User Control (Windows Forms) Visual C# Items
Component Class Visual C# Items
User Control (WPF) Visual C# Items
About Box (Windows Forms) Visual C# Items
ADO.NET Entity Data Model Visual C# Items
Application Configuration File Visual C# Items
Application Manifest File (Windows Only) Visual C# Items
Type: Visual C# Items
An empty class definition
ReadValueDemo
Add
Cancel

remember name of your class should follow PasCal casing

Your Interfaces, Enumerations, structure, methods inside classes all this should follow **pascal casing**

What is PascalCasing ?

---In every words 1st character should be capital