



VISUAL STUDIO

IDE FEATURES AND FUNCTIONS

Yunghans Irawan

yirawan@nus.edu.sg

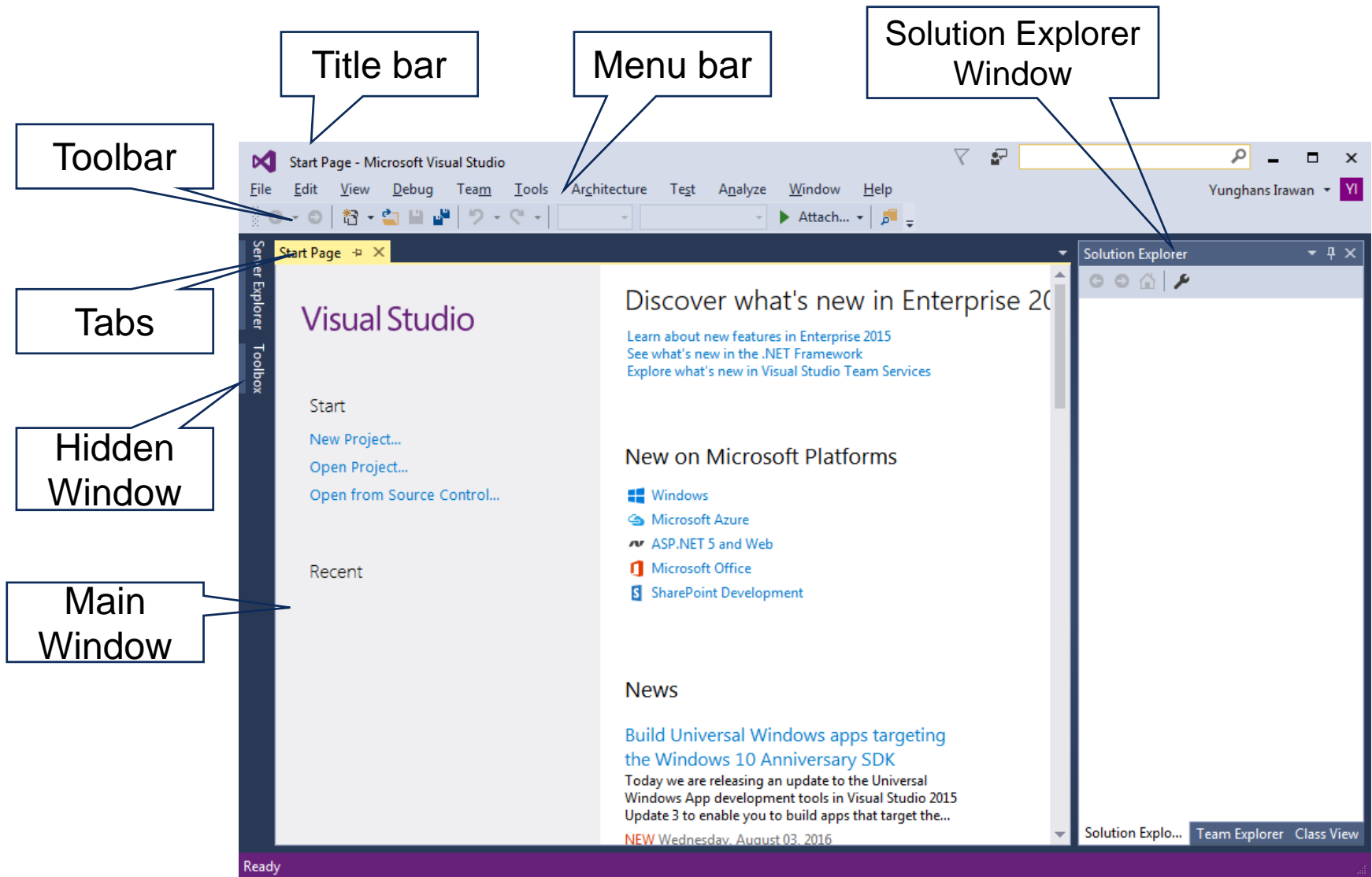
- Writing First Program
- Writing Second Program
- Debugging
- Other Features

- Integrated Development Environment
 - Editor
 - Visual Development Tools
 - Compiler
 - Debugger
- Purpose
 - Providing all these under one environment facilitates development
 - Tools improve productivity of developers
 - Visual Tools help in automating certain programming so coding is speeded up

- IDE stands for Integrated Development Environment
 - Integrated Development Environment consists of a number of elements:
 - Facility for managing Projects
 - Facility for Visual Form Developments
 - Facility for Program Coding, Compiling, Executing and Debugging
 - Facility for Visual Programming
 - Facility for reading Documentation and Context Sensitive Help
 - etc....
- IDE is where we put together our application.
 - Start Visual Basic and select a project type using the Integrated Development Environment or IDE which is an important part of Visual Basic.



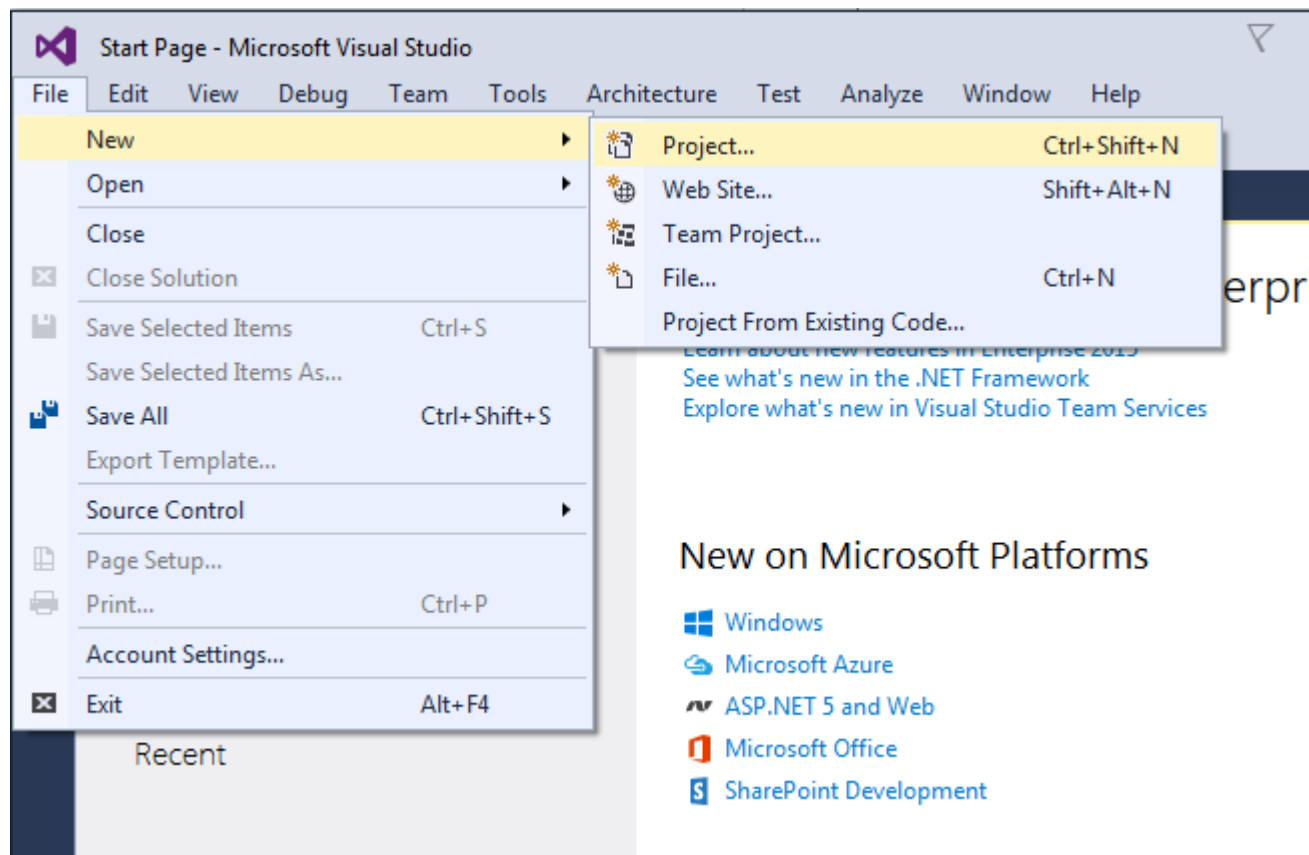
Microsoft Visual Studio





Writing the first program

- To create a new project, click on File > New > Project or press Ctrl+Shift+N





Writing the first program

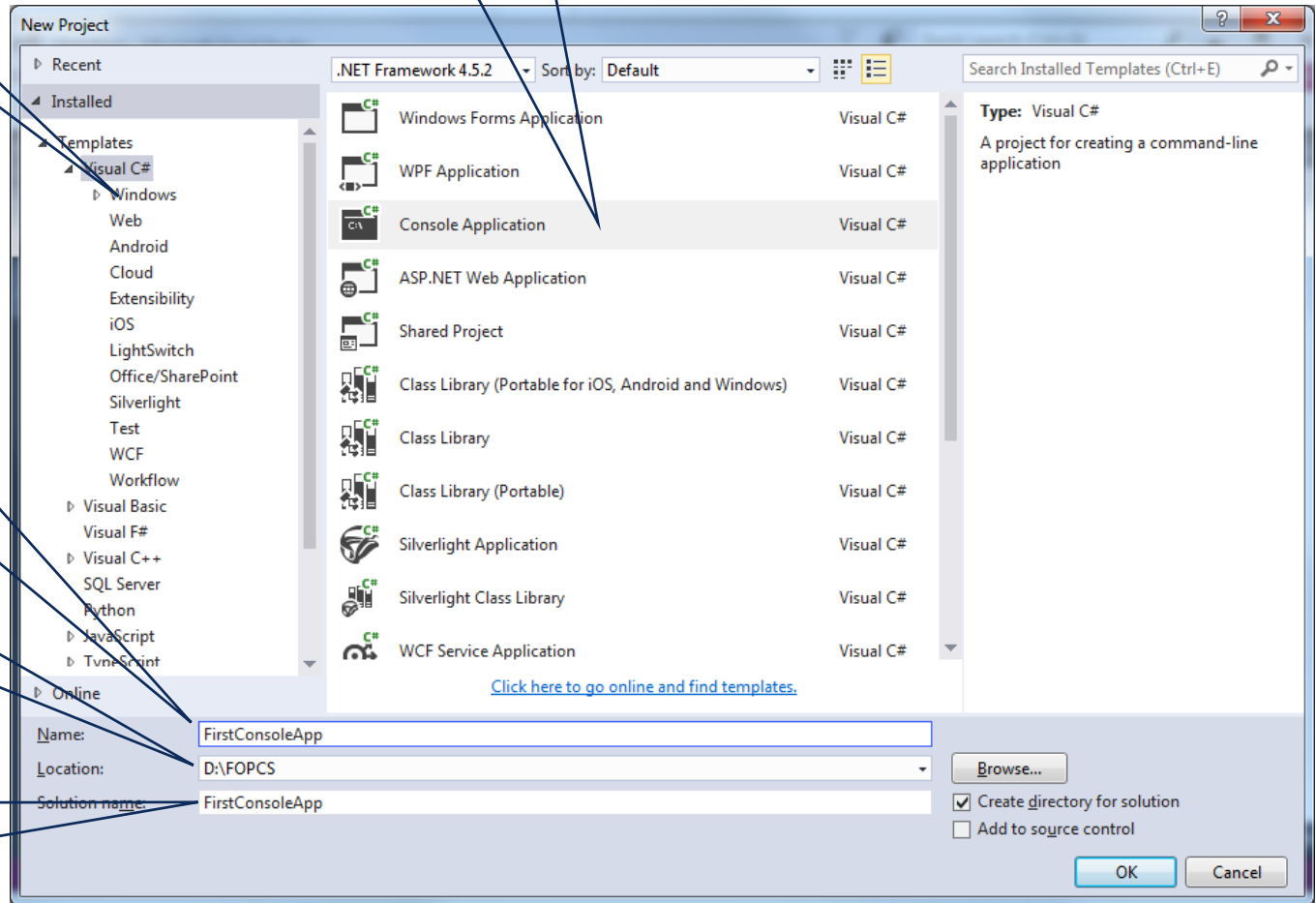
Choose C#

Console
Application

Name of the
project

Parent folder of
the solution
directory

Name of the
solution (group
of projects)





Namespaces used by
this code

Namespace
for your code

Dropdowns for
quick navigation

Content of your
solution and projects

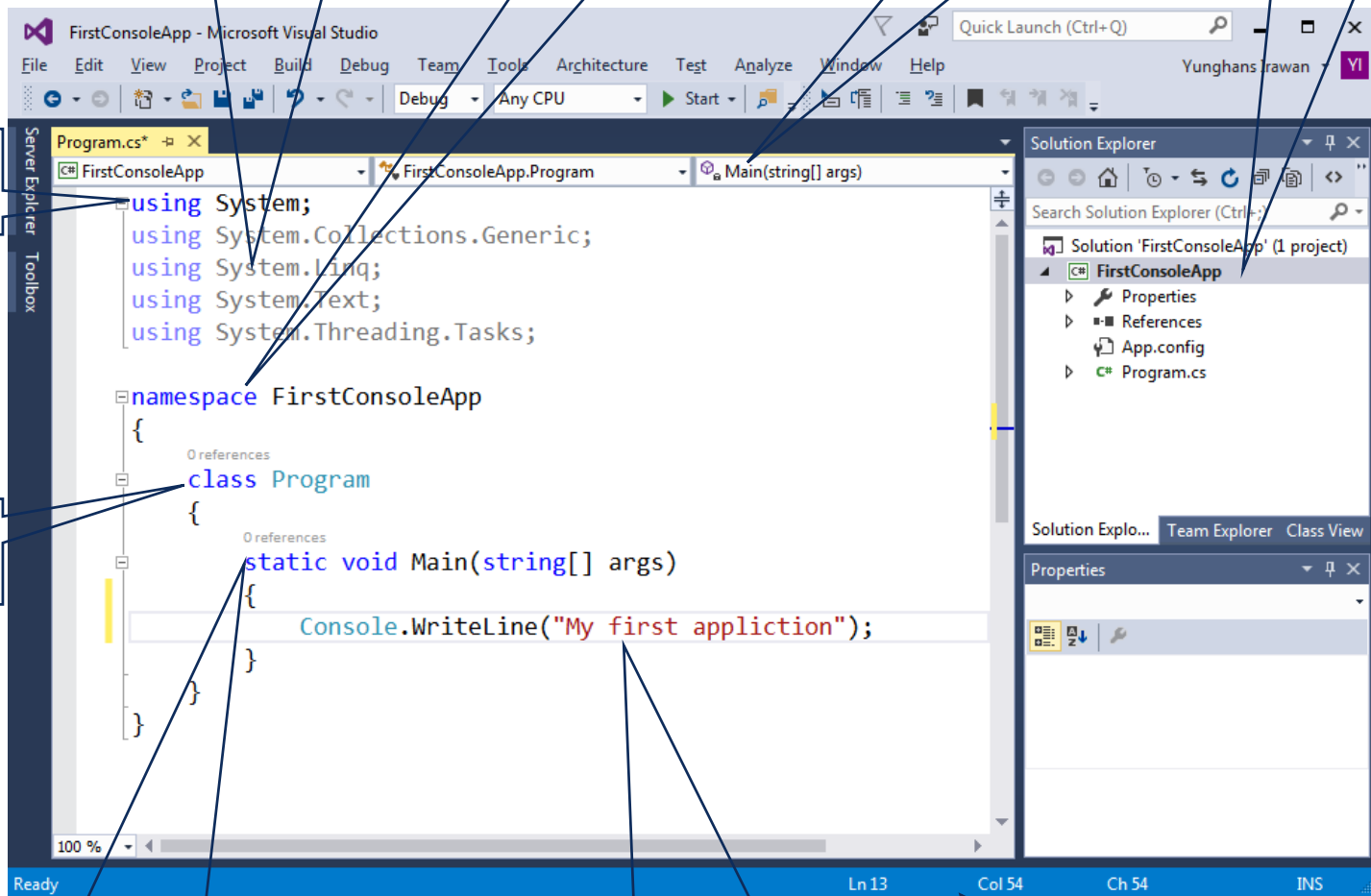
Expand and
collapse sign

Class
declaration

Main method is needed by
every startup class for
console application

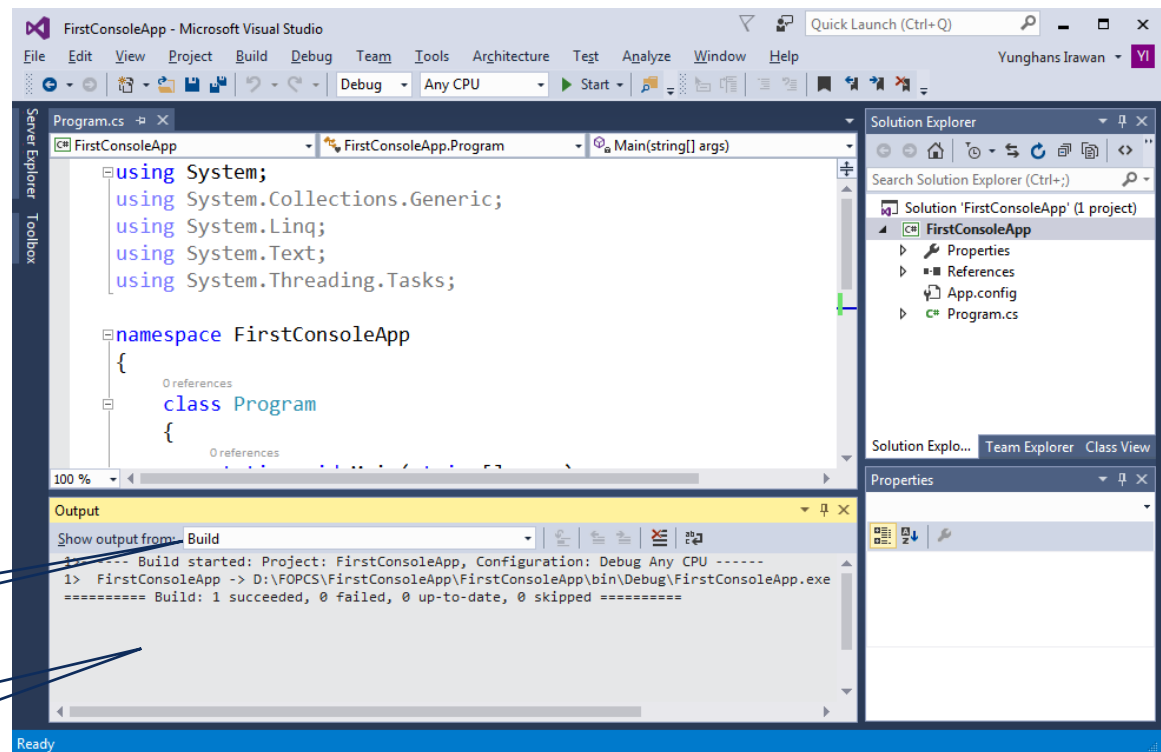
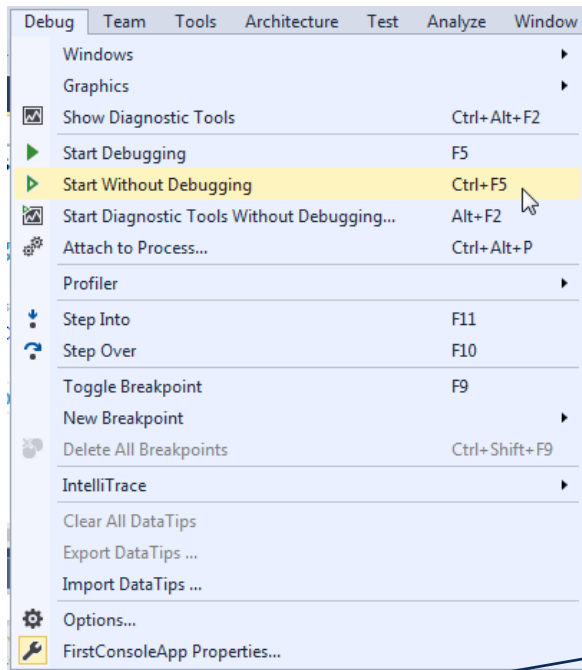
Method implementation

Location of the cursor



Running your application

- From the menu bar, select Debug > Start Without Debugging or press Ctrl+F5



The types of output shown

Output contains messages
related to the build process



Syntax Error

- If there's any syntax error, build will fail

Visual indicator of an error

Microsoft Visual Studio

There were build errors. Would you like to continue and run the last successful build?

Yes No

☐ Do not show this dialog again

Build failed

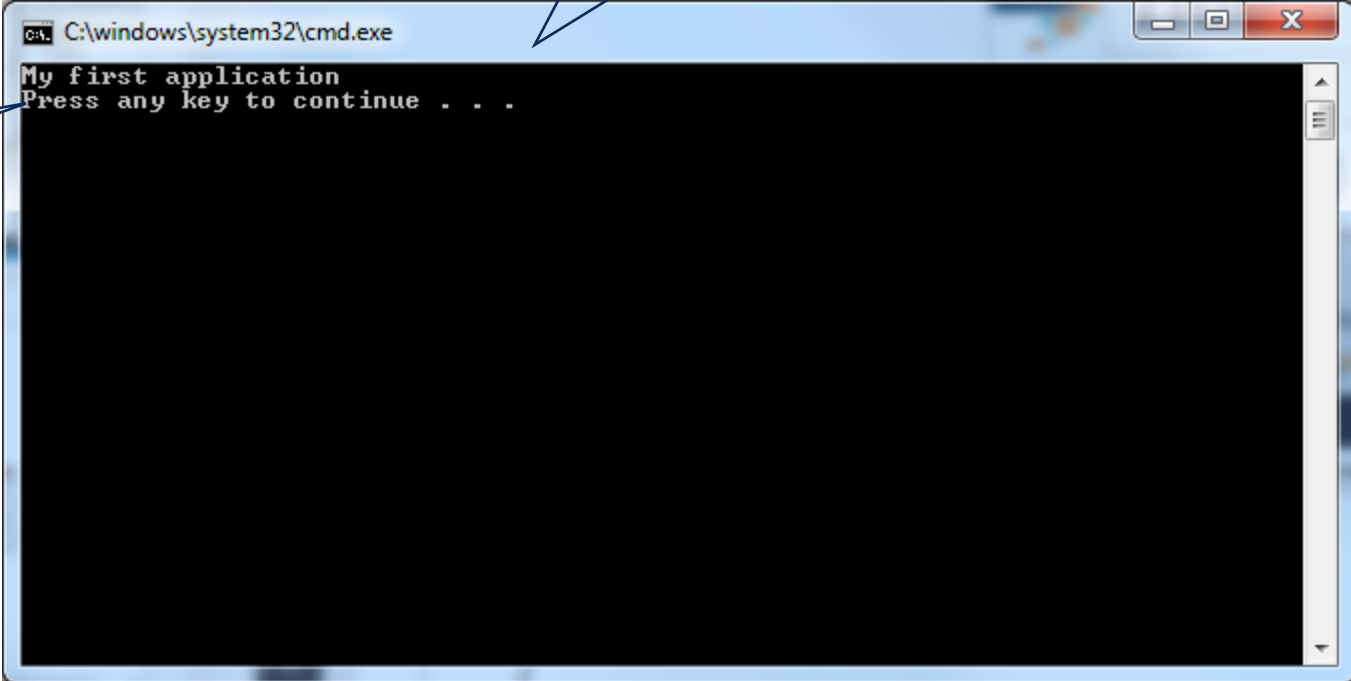
The scope of the list.
Useful when you have a long list

List of problems detected within the scope.

Code	Description	Project	File	Line	Suppression St...
CS1002	; expected	FirstConsoleApp	Program.cs	13	Active

Running your application

Input and output of a console application runs in a command line console



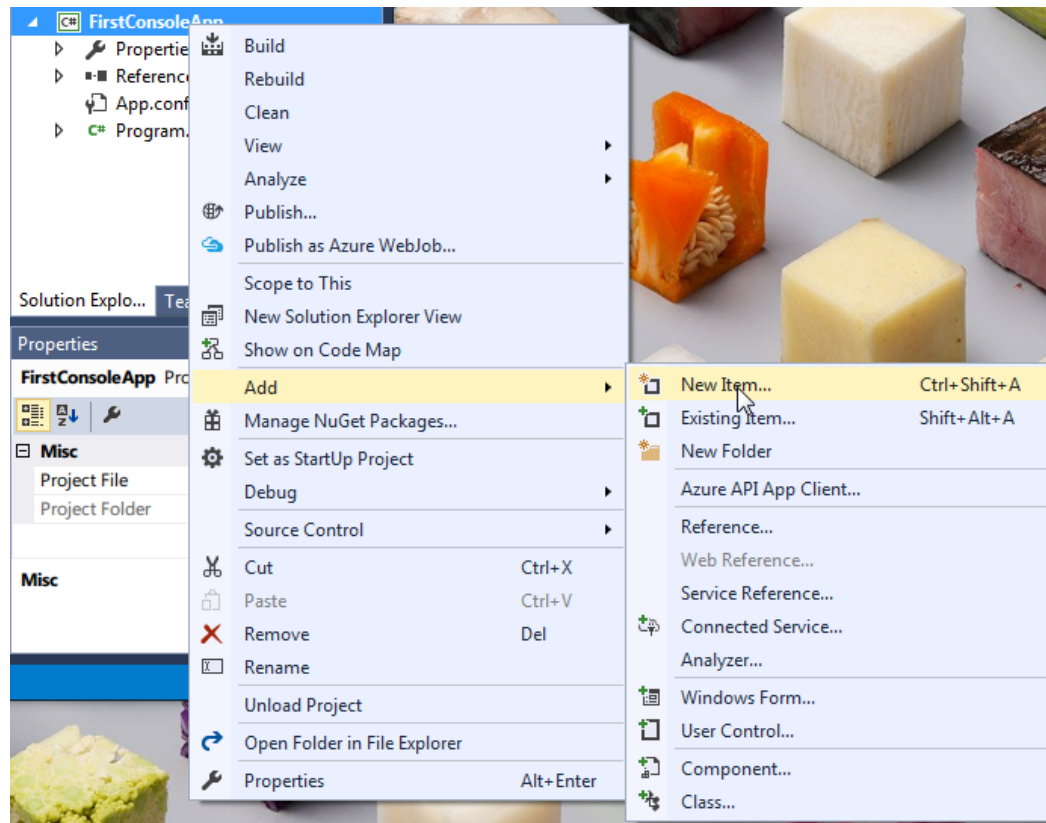
```
C:\windows\system32\cmd.exe
My first application
Press any key to continue . . .
```

The application will pause at the end of execution if you “Run Without Debugging”.

You can also add the pause command manually

Writing a second program

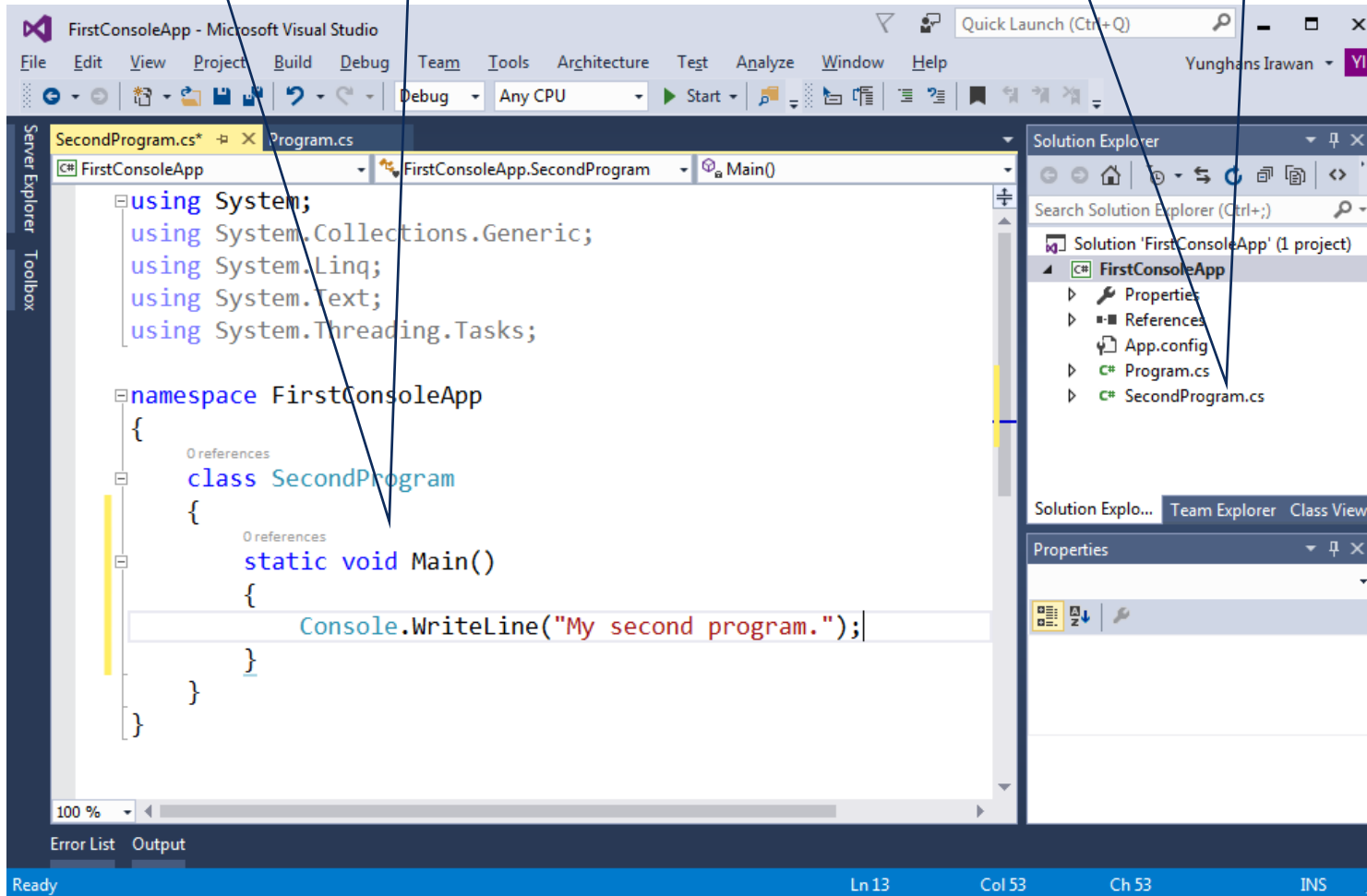
- You can write multiple programs in a single project – better if you have to write many simple programs
- Right click on the project, then select Add > New Item



Writing a second program

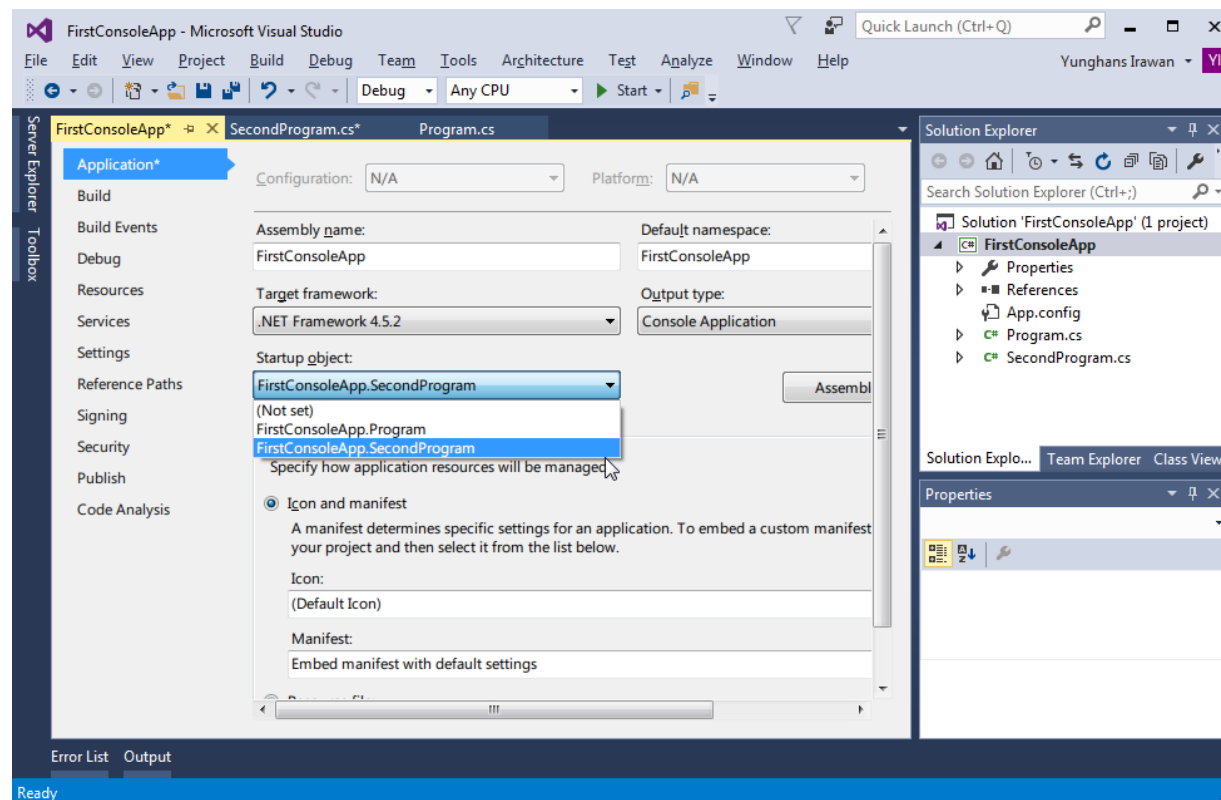
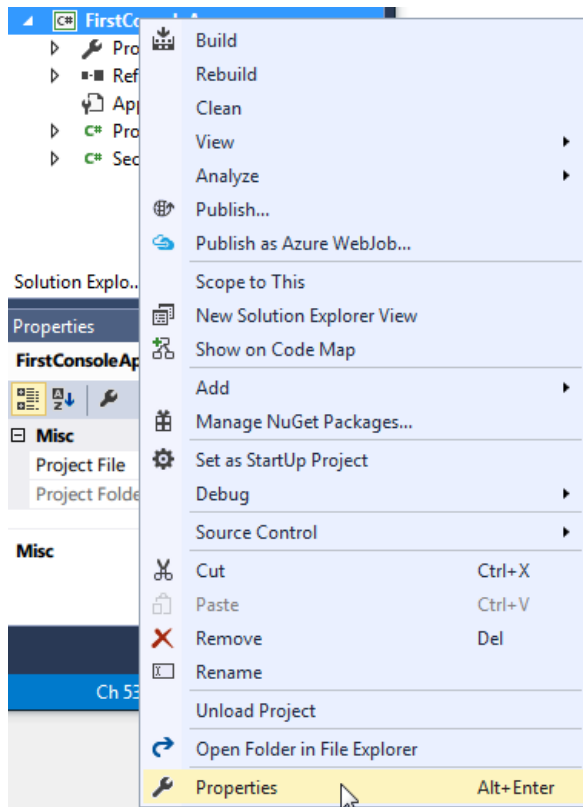
You need to write the content of the class.

The second program appears



Writing a second program

- If you try to run the project, VS will complain because there are two classes with Main method.
- Go the project properties by right clicking the project and select Properties and choose the Startup Class

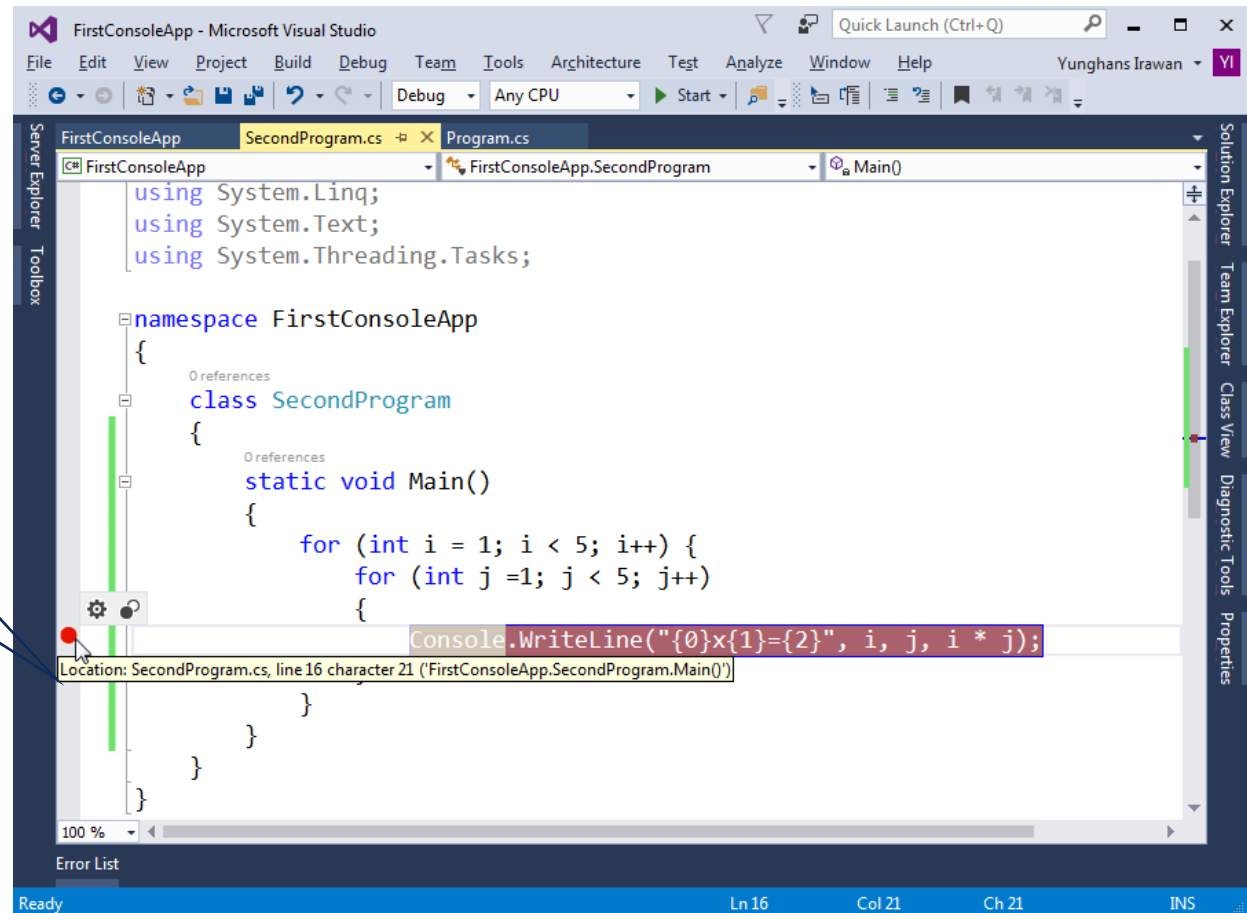




Debugging

- Change the second program as shown.
- Add a breakpoint by clicking on the grey area. A red dot will appear which indicates the breakpoint

Click on the side margin and you will see a red dot





Debugging

- To start debugging, press the Start button on the toolbar or press F5

Continue to run

Stop

Restart

Step
Into/Over/Out

The program is about
to run this line

The value of the
variables

Call stack: list of active
methods in the program

FirstConsoleApp (Debugging) - Microsoft Visual Studio

Process: [13284] FirstConsoleApp.vshost.exe Thread: [9496] Main Thread

```
static void Main()
{
    for (int i = 1; i < 5; i++) {
        for (int j = 1; j < 5; j++)
        {
            Console.WriteLine("{0}x{1}={2}", i, j, i*j);
        }
    }
}
```

Name	Value	Type
i	1	int
j	1	int

Name	Lang
FirstConsoleApp.exe!FirstConsoleApp.SecondProgram.Main() Line 16 C#	C#
[External Code]	

Ready Ln 16 Col 21 Ch 21 INS



Debugging

- Click Continue or “Step Into” or “Step Over” button to view the step by step execution of the code.

Continue	continue running – will pause if it reach a breakpoint
Step Into	Execute the line and stop at the next line. If this line calls a function, the next line would the be first line in that function
Step Over	Execute the line and stop at the next line. Ignore the implementation of any function in the current line

- Notice that the value of i and j changes as you debug the program



Code Editor View

The screenshot shows a code editor window titled "Program.cs*" with a "Start Page" tab. The code is a C# program for a console application. The code is as follows:

```
/* This program when executed prints the message
 * "Welcome to ISS"
 */

// Program written by Venkat

using System;
using System.Collections.Generic;
using System.Text;

namespace FirstConsoleApplication
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello");
            Console.WriteLine("Welcome to ISS");
        }
    }
}
```

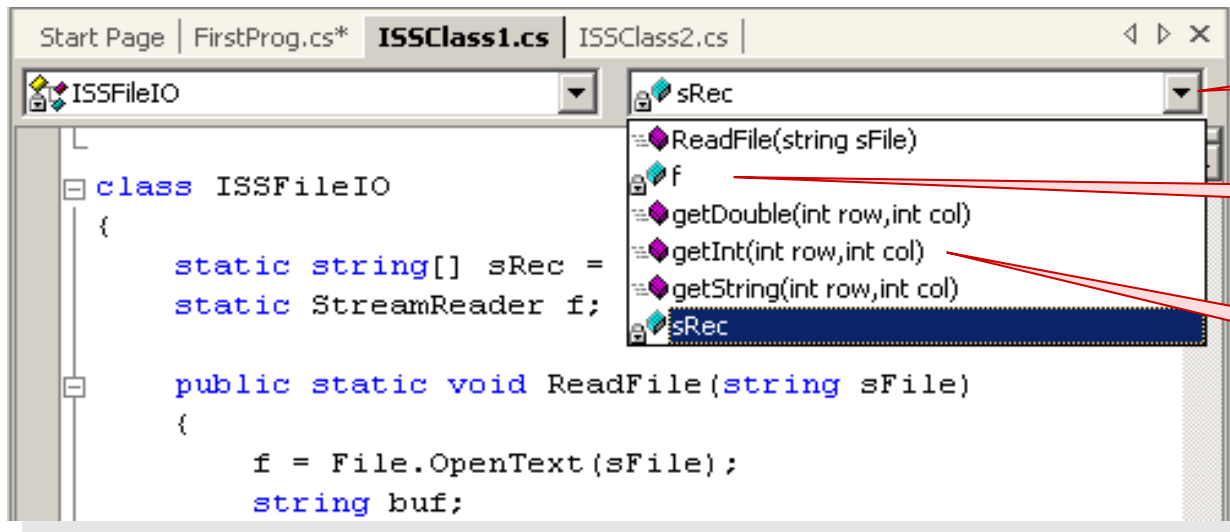
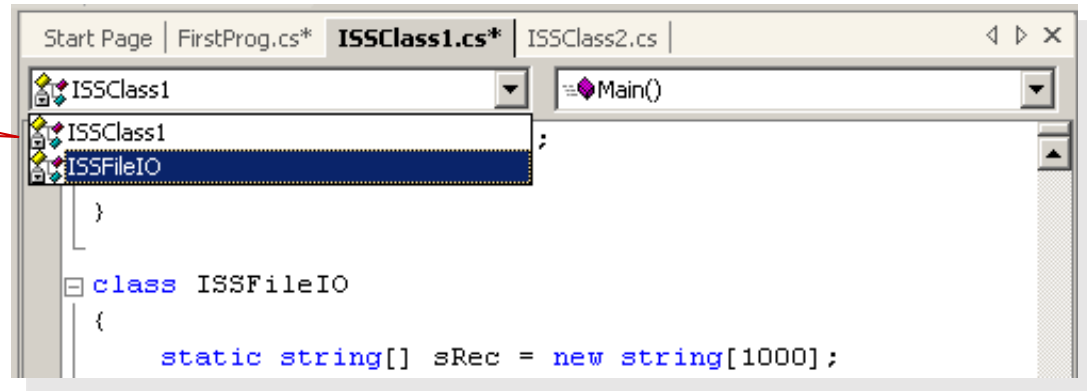
The code is annotated with several callouts:

- Page Tabs**: Points to the "Program.cs*" and "Start Page" tabs at the top of the editor.
- Types (Classes)**: Points to the `class Program` declaration.
- Expand / Collapse**: Points to the expand/collapse icons in the left margin.
- Keywords (Dark Blue)**: Points to the `using`, `namespace`, and `class` keywords.
- Comments (Green)**: Points to the multi-line comment at the top of the code.
- Members (Methods)**: Points to the `Main` method signature.
- Error (Red Underline)**: Points to the red squiggly line under the closing quote of the second `Console.WriteLine` call.
- Code Formatting Auto-Aligned**: Points to the indentation of the code lines.



List Boxes in Editor

Types List
(Class)



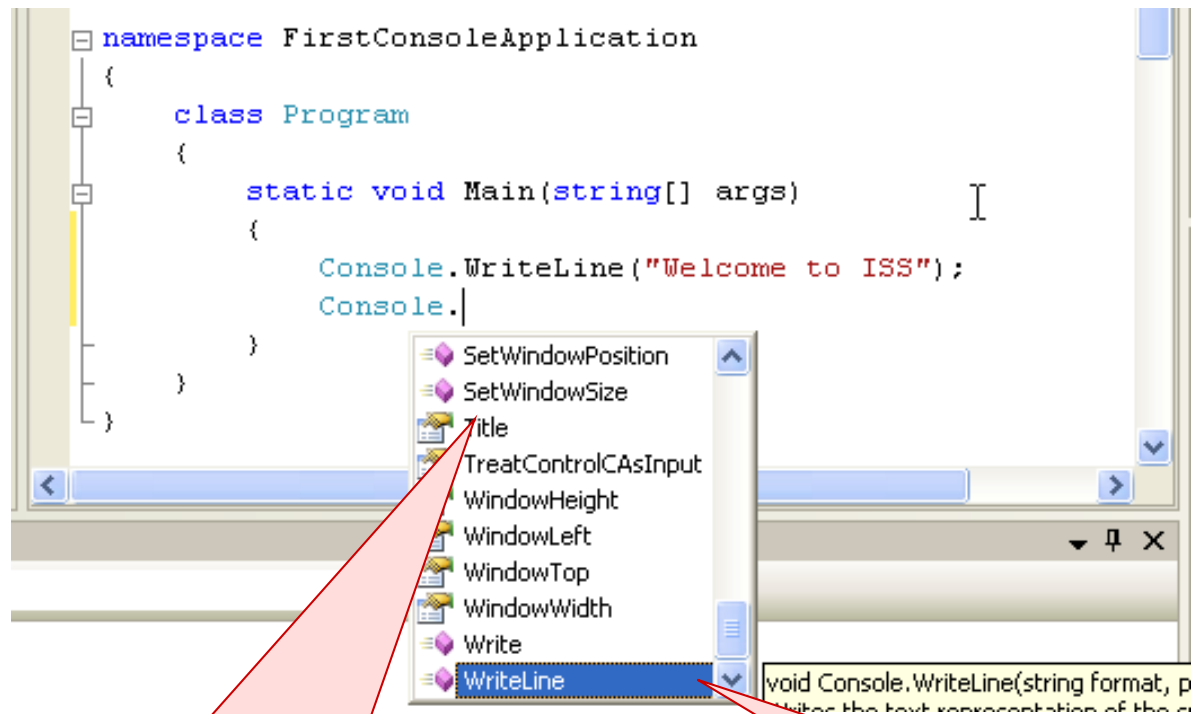
Member List

Variable

Method

- What is Intellisense?
 - It is a editor feature, that helps the developer with prompts, lookup and other visual features.
 - It improves the programmers editing productivity
- What are the features?
 - List Members
 - Parameter Info
 - Quick Info
 - Complete Word
 - Automatic Brace Matching

Intellisense: List Members



List of all members in Console Class
(Drops down automatically when typing a dot)

Also listed the familiar
Write & WriteLine !

Intellisense: List Members

```
Console.WriteLine (
```

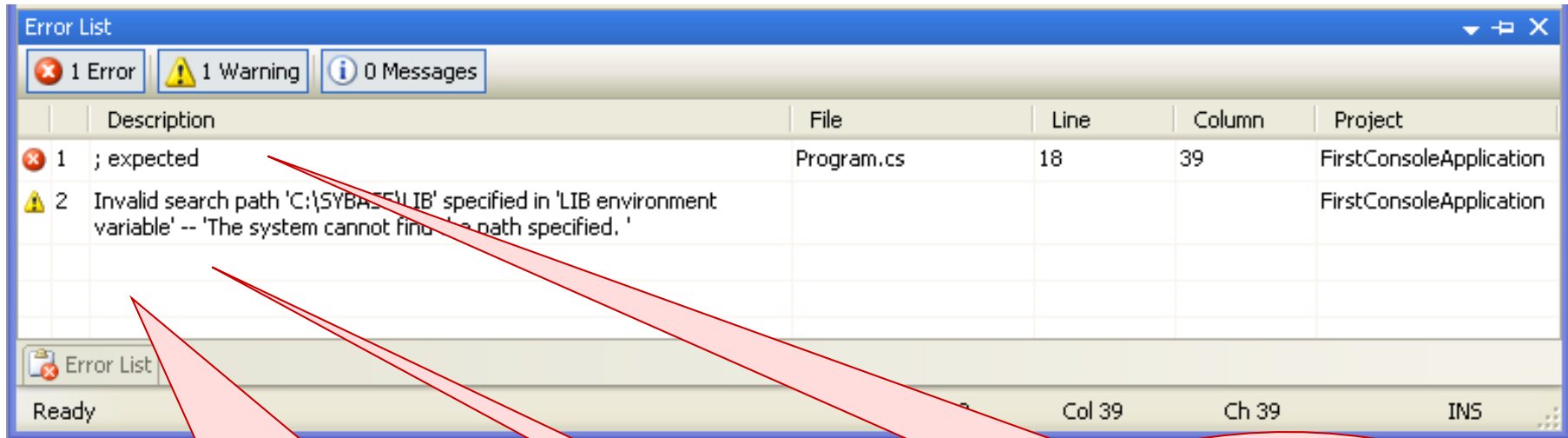
▲ 15 of 19 ▼ void Console.WriteLine (**string format**, params object[] arg)
format: The format string.

Parameter Info:
How many and what type.

Currently Editing Parameter shown in bold



Task list Window

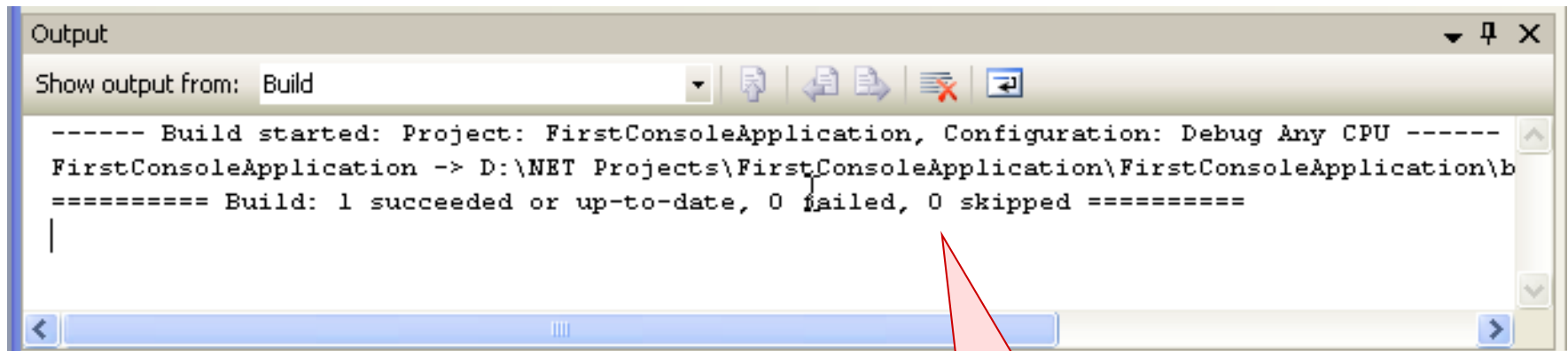


Task List: Build (i.e. compilation errors are listed here one per line. Double Clicking a line takes you to appropriate line in code.

This error states that a semi-colon is expected but not found.

Warnings in yellow

Output Window



The screenshot shows the 'Output' window in a development environment. The title bar says 'Output'. Below it, a dropdown menu is set to 'Build'. The main text area contains the following output:

```
----- Build started: Project: FirstConsoleApplication, Configuration: Debug Any CPU -----  
FirstConsoleApplication -> D:\NET Projects\FirstConsoleApplication\FirstConsoleApplication\b  
===== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped =====  
|
```

The window has a scrollbar on the right and a status bar at the bottom.

Output Messages: Like Compiler message, Print Message of Window Application etc. are displayed here

- Standard Project Types for C#
 - Windows Application
 - Class Library
 - Windows Control Library
 - ASP .Net Web Application
 - ASP .Net Web Service
 - Web Control Library
 - Console Application
 - Windows Service
 - Empty Project
 - ConsoleApplication(ISS-RV) *

- Windows Application
 - This project is used to create Windows Application driven by Windows Forms
 - We would use this type of Projects in GUI module
- Class Library
 - This type of project helps in creating .Net Class Libraries which could be deployed independently and used across projects.
 - These are usually dll files which are not stand alone applications.
 - We would be developing Class Libraries when studying Enterprise Software Development lessons.

- **ASP .Net Web Application**
 - This project is used to create Web based Application using the ASP .Net development platform.
 - We would use this type of Projects in Internet module lessons
- **Console Application**
 - These type of projects would be used for Console Applications
 - All the applications that you are currently learning are Console Applications.
 - This could be used for your current lessons

- Revision of Key Concepts
- Discussion on Workshop
- Questions and Answer
- Assignments and Home Work