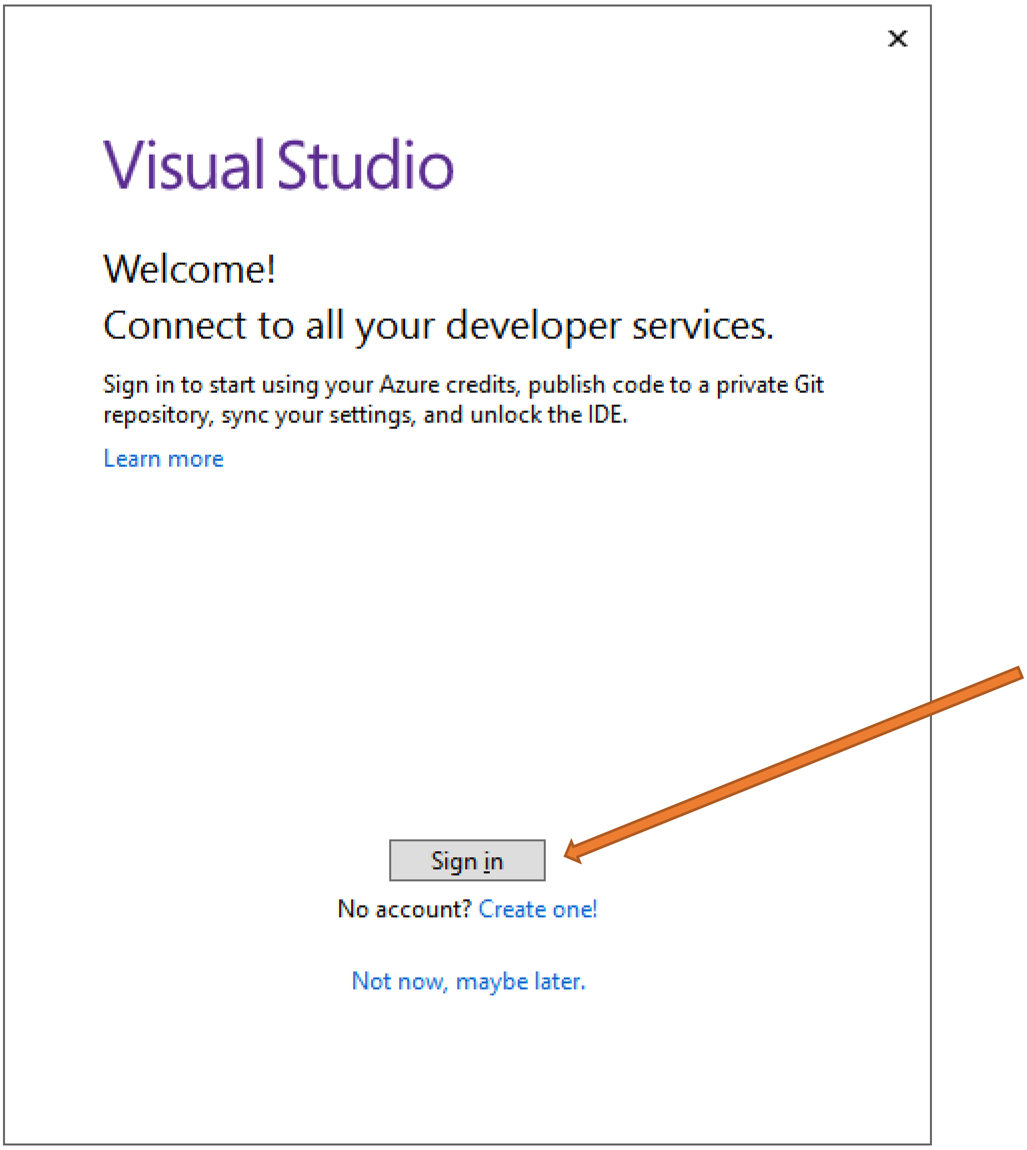
CM1205

ASSEMBLER LAB 1

We are going to be using Visual Studio environment to write, assemble and run our assembly language programs.

Before we can start to learn assembler we need to understand how to setup a project in Visual Studio.

1) On the lab machine go to start button and type/search for **Visual Studio 2017.**
2) You may see this screen at the beginning. If so, click "Sign in" and sign in using your **Cardiff University email address**. It will then redirect to the Cardiff University sign in page. Follow the instructions to sign in from there.



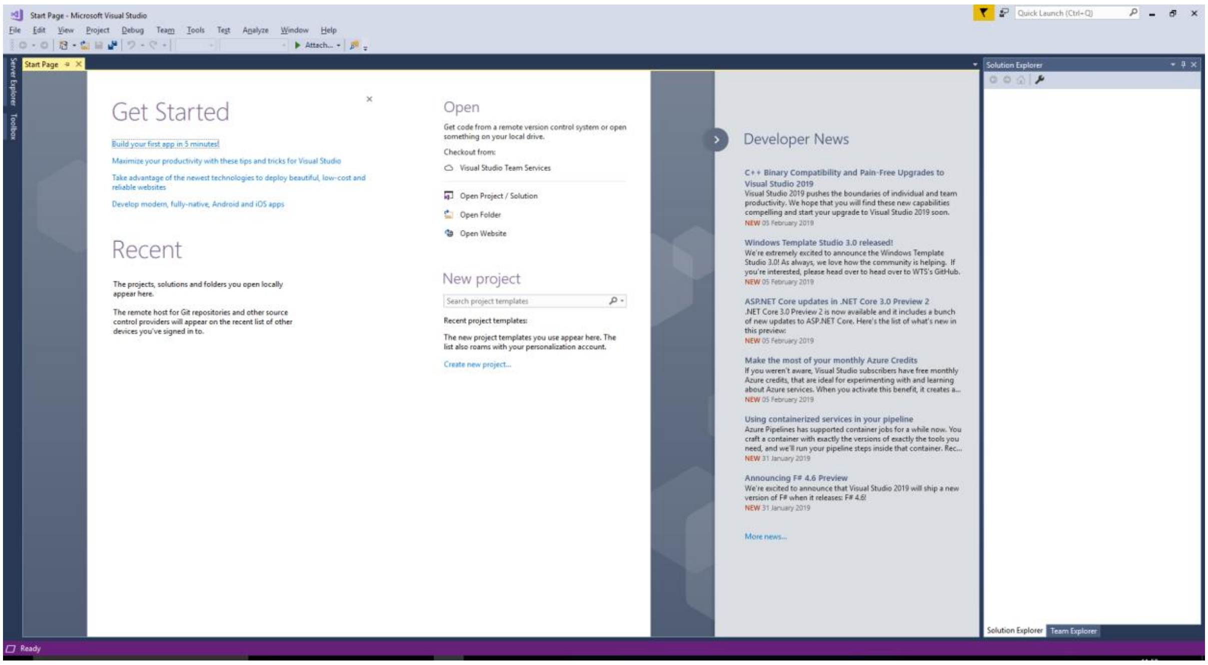
Visual Studio

Welcome!

Connect to all your developer services.

Sign in to start using your Azure credits, publish code to a private Git repository, sync your settings, and unlock the IDE.

3) You should then see the following startup screen


Start Page - Microsoft Visual Studio
File Edit View Project Debug Team Tools Test Analyze Window Help
Attach...
Quick Launch (Ctrl+Q)
Start Page
Server Explorer
Toolbox
Solution Explorer
Team Explorer
Ready

Get Started

Build your first app in 5 minutes!

Maximize your productivity with these tips and tricks for Visual Studio

Take advantage of the newest technologies to deploy beautiful, low-cost and reliable websites

Develop modern, fully-native, Android and iOS apps

Recent

The projects, solutions and folders you open locally appear here.

The remote host for Git repositories and other source control providers will appear on the recent list of other devices you've signed in to.

Open

Get code from a remote version control system or open something on your local drive.

Checkout from:

Visual Studio Team Services

Open Project / Solution

Open Folder

Open Website

New project

Search project templates

Recent project templates:

The new project templates you use appear here. The list also roams with your personalization account.

Create new project...

Developer News

C++ Binary Compatibility and Pain-Free Upgrades to Visual Studio 2019

Visual Studio 2019 pushes the boundaries of individual and team productivity. We hope that you will find these new capabilities compelling and start your upgrade to Visual Studio 2019 soon.

NEW 05 February 2019

Windows Template Studio 3.0 released!

We're extremely excited to announce the Windows Template Studio 3.0! As always, we love how the community is helping. If you're interested, please head over to head over to WTS's GitHub.

NEW 05 February 2019

ASP.NET Core updates in .NET Core 3.0 Preview 2

.NET Core 3.0 Preview 2 is now available and it includes a bunch of new updates to ASP.NET Core. Here's the list of what's new in this preview:

NEW 05 February 2019

Make the most of your monthly Azure Credits

If you weren't aware, Visual Studio subscribers have free monthly Azure credits, that are ideal for experimenting with and learning about Azure services. When you activate this benefit, it creates a...

NEW 05 February 2019

Using containerized services in your pipeline

Azure Pipelines has supported container jobs for a while now. You craft a container with exactly the versions of exactly the tools you need, and we'll run your pipeline steps inside that container. Rec...

NEW 31 January 2019

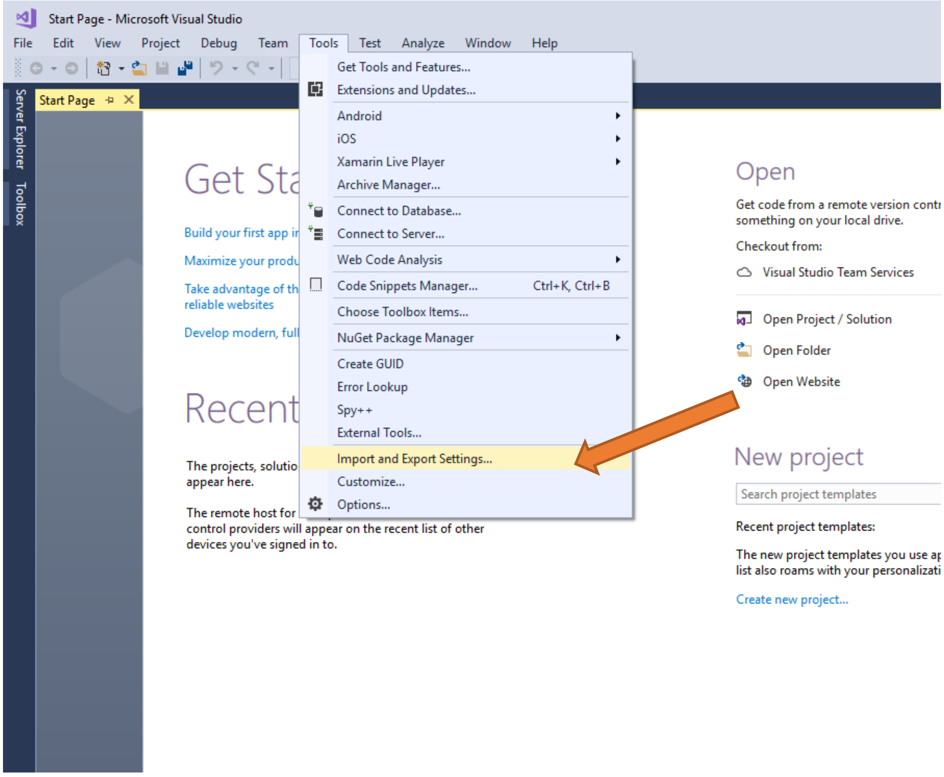
Announcing F# 4.6 Preview

We're excited to announce that Visual Studio 2019 will ship a new version of F# when it releases: F# 4.6!

NEW 31 January 2019

More news...

4) INITIAL SETUP. (**You will only need to do this once.**)
From the **"TOOLS"** menu – select "**Import and Export Settings":**



Get Sta

Build your first app ir

Maximize your produ

Take advantage of th reliable websites

Develop modern, full

Recent

The projects, solutio appear here.

The remote host for control providers will appear on the recent list of other devices you've signed in to.

Open

Get code from a remote version contr something on your local drive.

Checkout from:

Visual Studio Team Services

Open Project / Solution

Open Folder

Open Website

New project

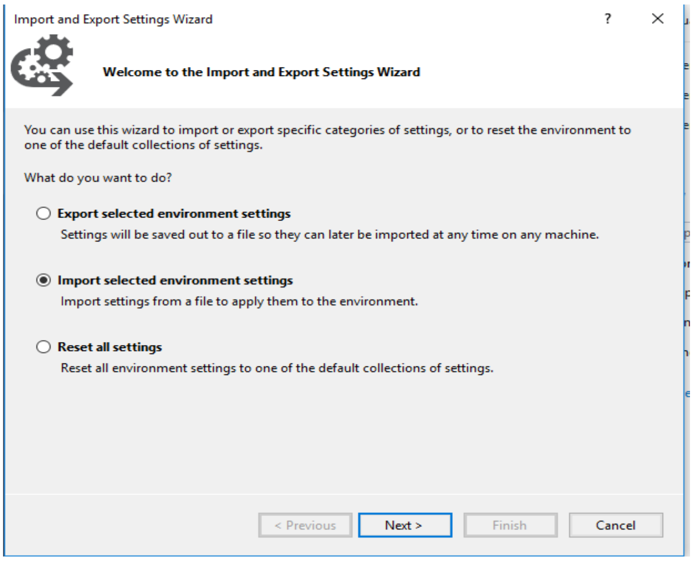
Search project templates

Recent project templates:

The new project templates you use ap list also roams with your personalizati

Create new project...

5) Select the “**Import selected environment settings**” radio button



Import and Export Settings Wizard

Welcome to the Import and Export Settings Wizard

You can use this wizard to import or export specific categories of settings, or to reset the environment to one of the default collections of settings.

What do you want to do?

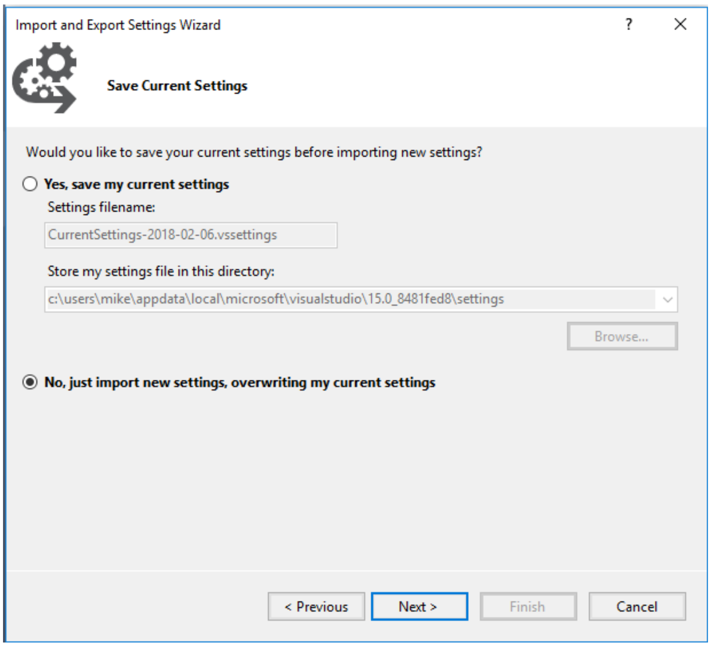
○ **Export selected environment settings**
Settings will be saved out to a file so they can later be imported at any time on any machine.

● **Import selected environment settings**
Import settings from a file to apply them to the environment.

○ **Reset all settings**
Reset all environment settings to one of the default collections of settings.

Then select “**Next**”

6) Select “**No, just import…**” radio button



Import and Export Settings Wizard

Save Current Settings

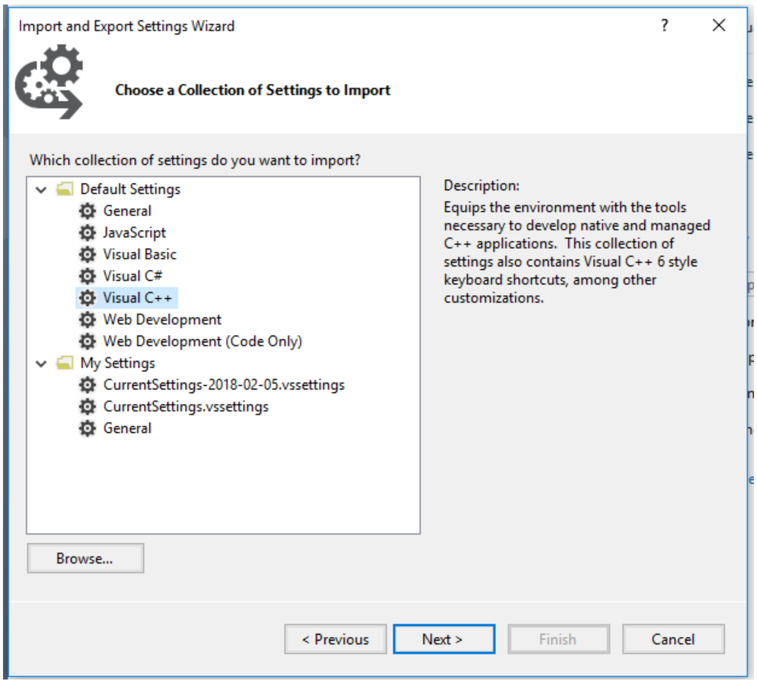
Would you like to save your current settings before importing new settings?

○ **Yes, save my current settings**
Settings filename:
CurrentSettings-2018-02-06.vssettings
Store my settings file in this directory:
c:\users\mike\appdata\local\microsoft\visualstudio\15.0_8481fed8\settings

● **No, just import new settings, overwriting my current settings**

Then select “**Next**”

7) Select “**Visual C++**” from the default settings list and click the “**Next**” button.



Import and Export Settings Wizard

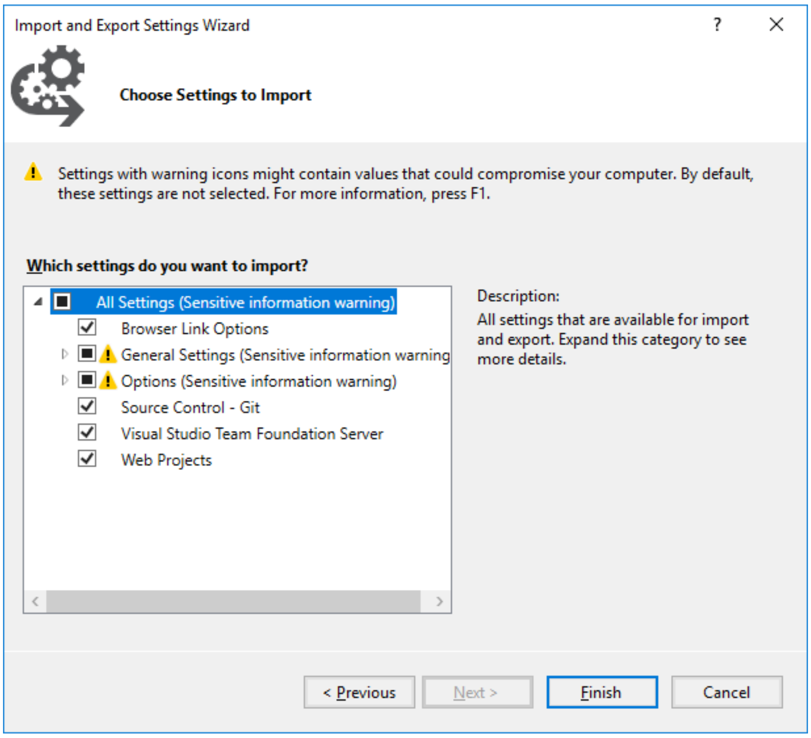
Choose a Collection of Settings to Import

Which collection of settings do you want to import?

- Default Settings
  - General
  - JavaScript
  - Visual Basic
  - Visual C#
  - Visual C++
  - Web Development
  - Web Development (Code Only)
- My Settings
  - CurrentSettings-2018-02-05.vssettings
  - CurrentSettings.vssettings
  - General

Description:
Equips the environment with the tools necessary to develop native and managed C++ applications. This collection of settings also contains Visual C++ 6 style keyboard shortcuts, among other customizations.

8) On the next screen just click “**Finish**” (ignore the warnings) and then click the “**close**” button:



Import and Export Settings Wizard

Choose Settings to Import

Settings with warning icons might contain values that could compromise your computer. By default, these settings are not selected. For more information, press F1.

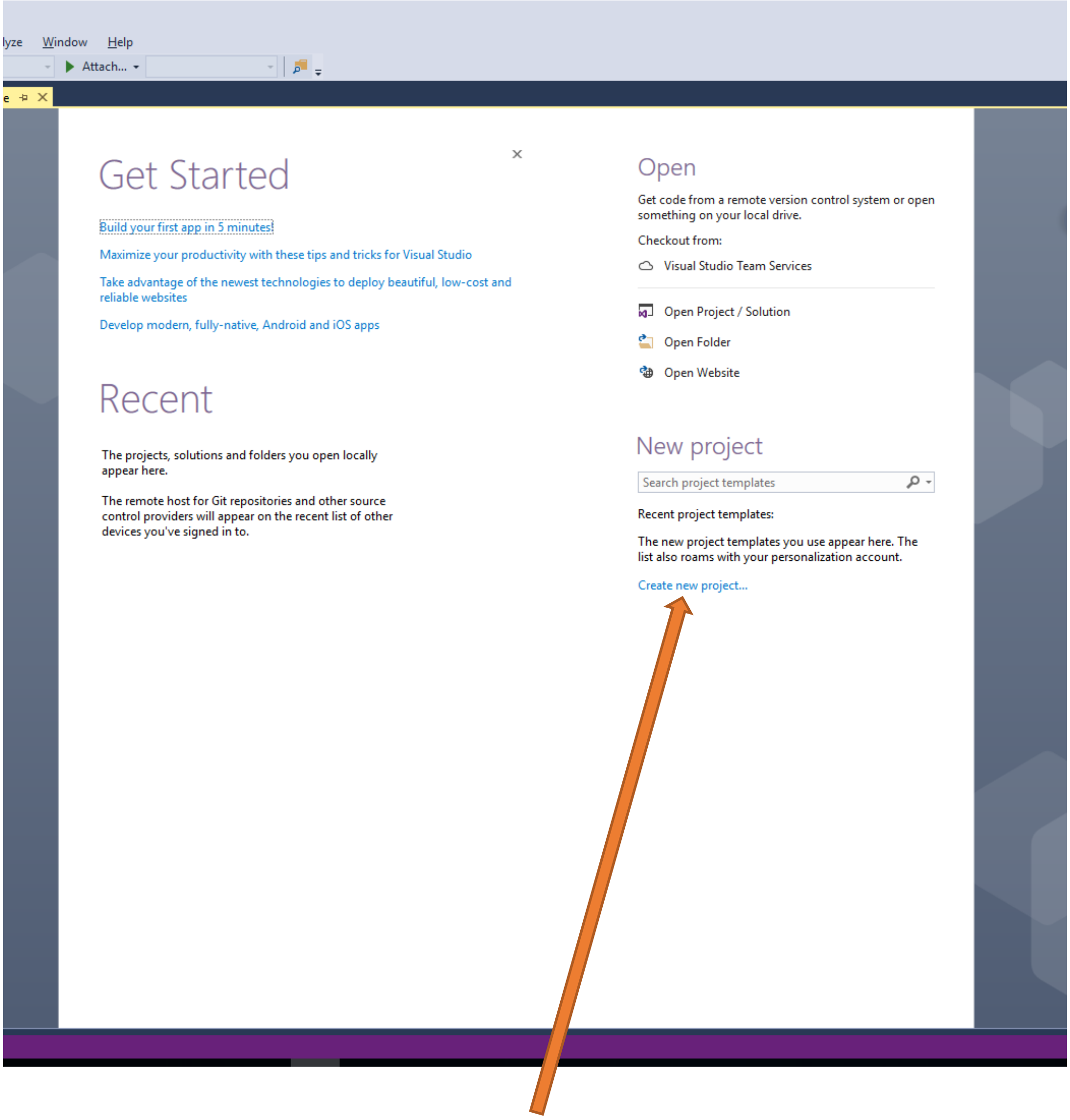
Which settings do you want to import?

- ■ All Settings (Sensitive information warning)
  - ☑ Browser Link Options
  - ■ ⚠ General Settings (Sensitive information warning
  - ■ ⚠ Options (Sensitive information warning)
  - ☑ Source Control - Git
  - ☑ Visual Studio Team Foundation Server
  - ☑ Web Projects

Description:
All settings that are available for import and export. Expand this category to see more details.

9) Creating our first program and running it.

You should now be back at the **“Get Started”** page again



Get Started

Build your first app in 5 minutes!

Maximize your productivity with these tips and tricks for Visual Studio

Take advantage of the newest technologies to deploy beautiful, low-cost and reliable websites

Develop modern, fully-native, Android and iOS apps

Recent

The projects, solutions and folders you open locally appear here.

The remote host for Git repositories and other source control providers will appear on the recent list of other devices you've signed in to.

Open

Get code from a remote version control system or open something on your local drive.

Checkout from:

Visual Studio Team Services

Open Project / Solution

Open Folder

Open Website

New project

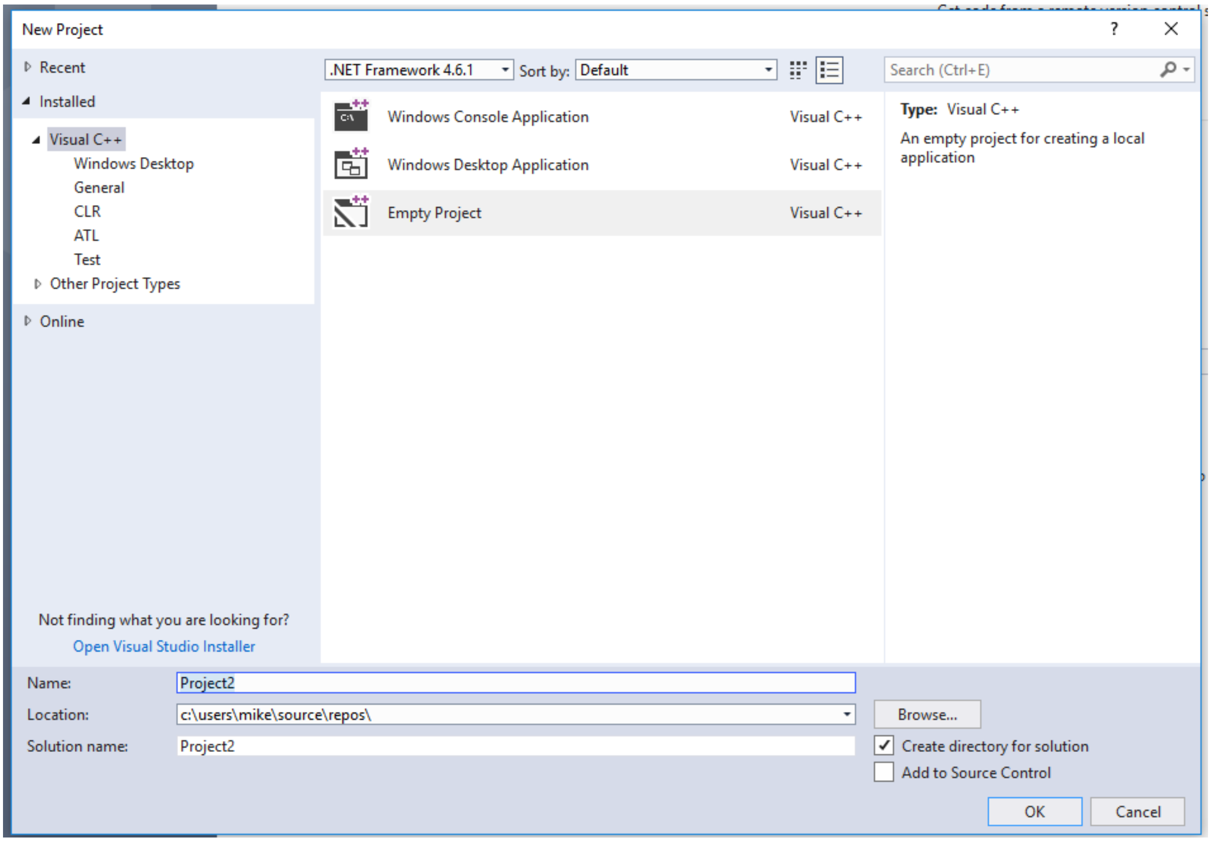
Search project templates

Recent project templates:

The new project templates you use appear here. The list also roams with your personalization account.

Create new project...

Select **“Create new project”** by clicking the link shown on the “Get Started” page, or you can achieve the same thing by going to “File” in the top-left menu, then “New”, and then “Project”



New Project

.NET Framework 4.6.1

Sort by: Default

Search (Ctrl+E)

- Recent
- Installed
  - Visual C++
    - Windows Desktop
    - General
    - CLR
    - ATL
    - Test
  - Other Project Types
- Online

Windows Console Application	Visual C++
Windows Desktop Application	Visual C++
Empty Project	Visual C++

Type: Visual C++

An empty project for creating a local application

Not finding what you are looking for?

Open Visual Studio Installer

Name: Project2

Location: c:\users\mike\source\repos\

Solution name: Project2

☑ Create directory for solution

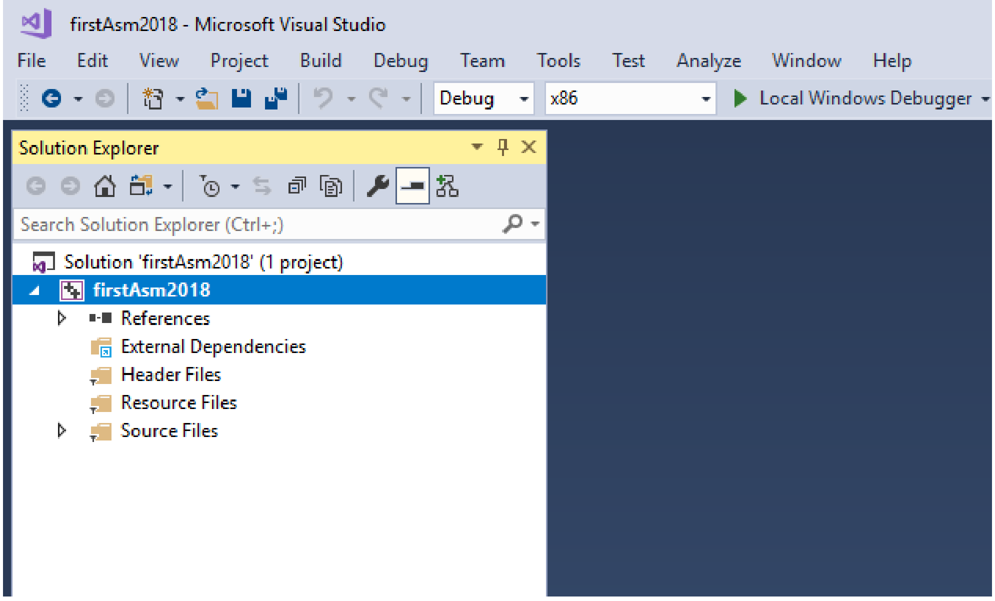
□ Add to Source Control

Select **“Empty Project”**

In the **“Name”** field enter a suitable name for your project. I chose the name firstAsm2018 (I like to live in the past).

Click **“OK”**

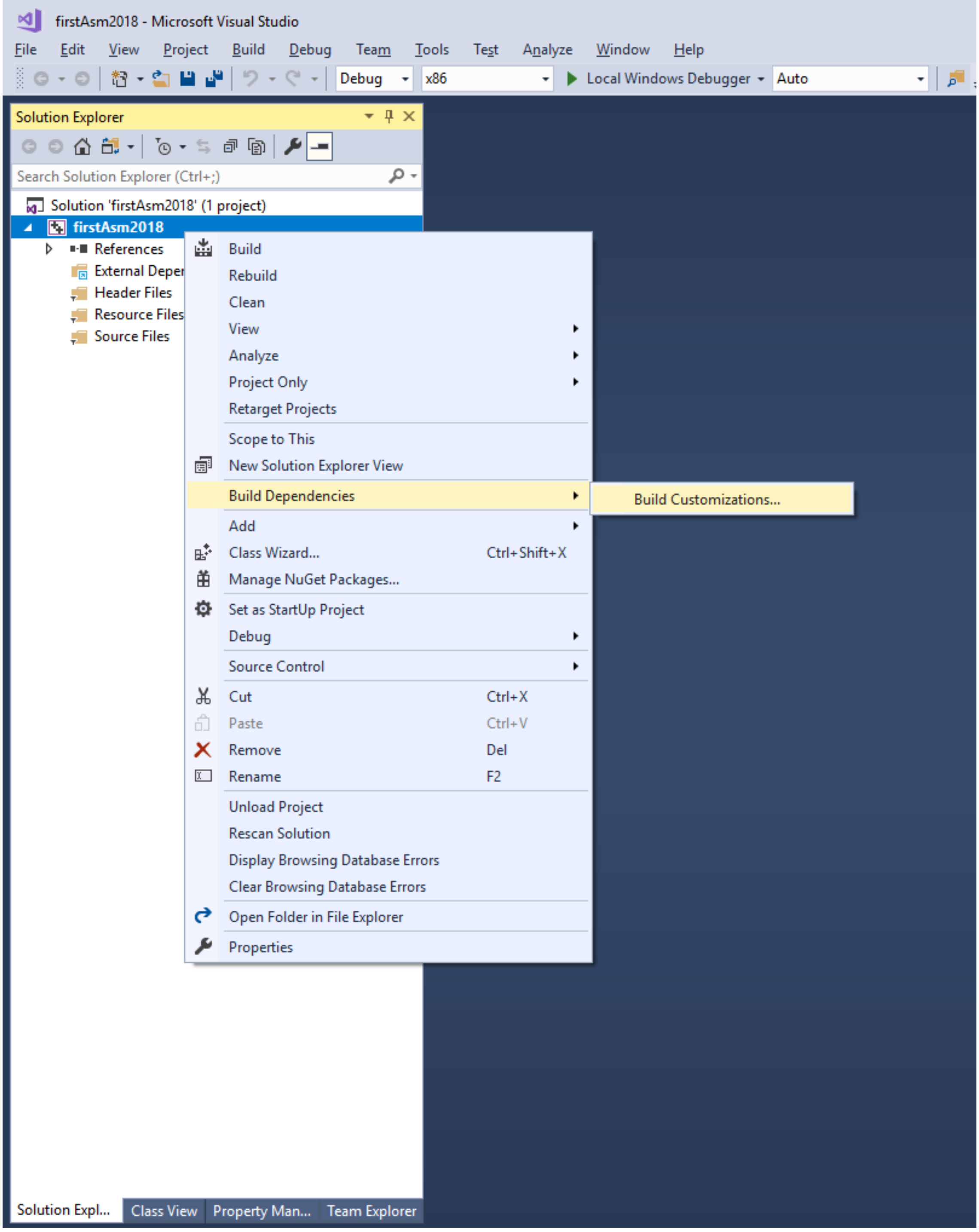
10) A new project will now be created. You should see the Project appear in the **“Solution Explorer”** pane.



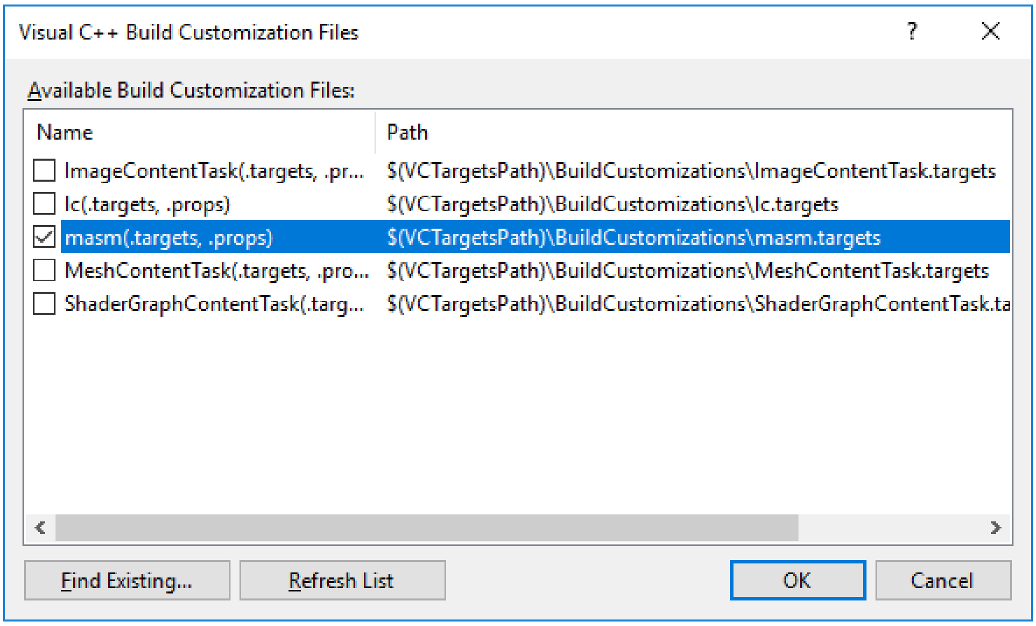
firstAsm2018 - Microsoft Visual Studio

- Solution 'firstAsm2018' (1 project)
  - firstAsm2018
    - References
    - External Dependencies
    - Header Files
    - Resource Files
    - Source Files

Right click on the project name, hover over **“Build Dependencies”** and select **Build Customizations.**



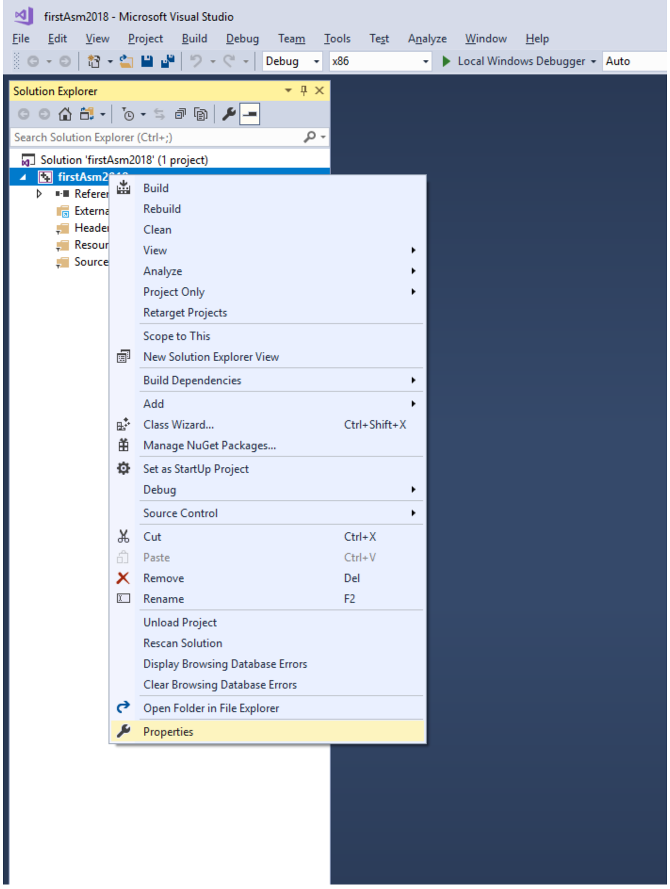
You should see something similar to the following:


Visual C++ Build Customization Files
?
Available Build Customization Files:

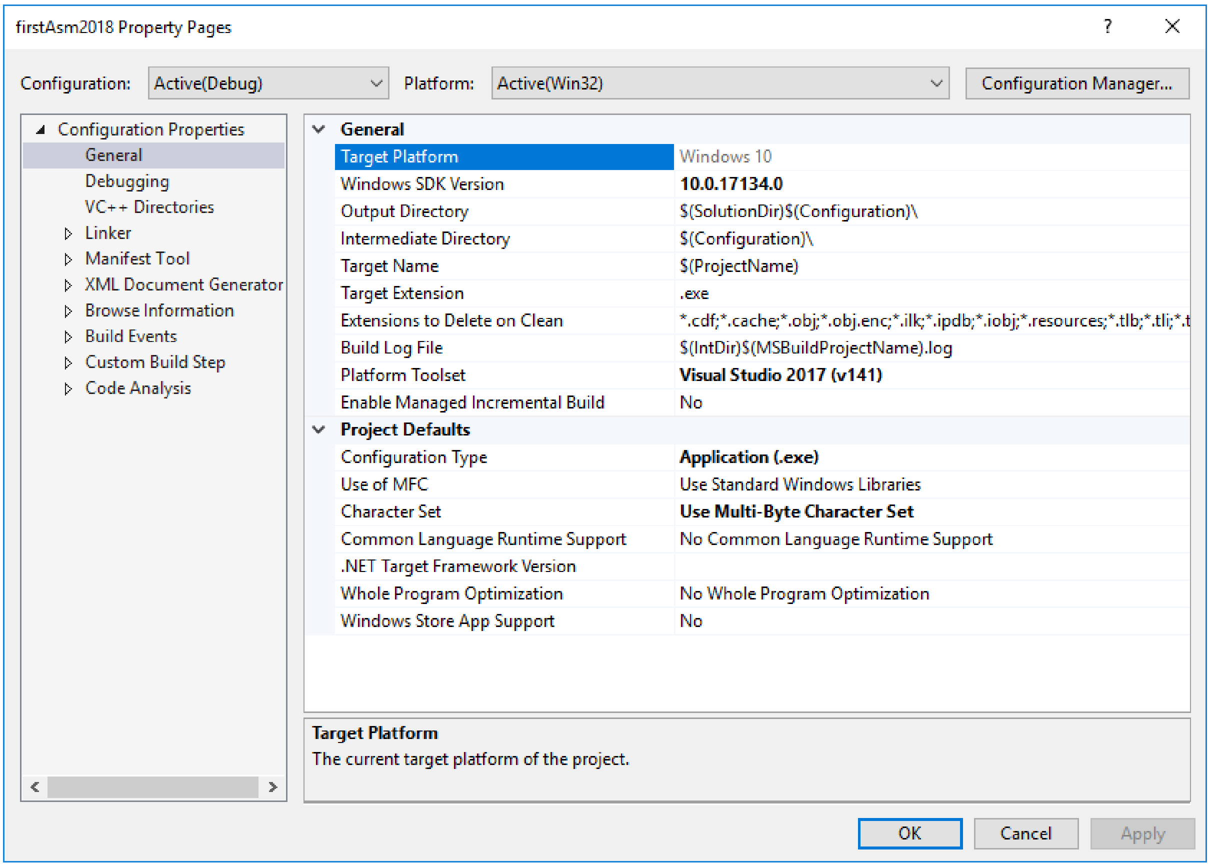
Name	Path
☐ ImageContentTask(.targets, .pr...	$(VCTargetsPath)\BuildCustomizations\ImageContentTask.targets
☐ lc(.targets, .props)	$(VCTargetsPath)\BuildCustomizations\lc.targets
☑ masm(.targets, .props)	$(VCTargetsPath)\BuildCustomizations\masm.targets
☐ MeshContentTask(.targets, .pro...	$(VCTargetsPath)\BuildCustomizations\MeshContentTask.targets
☐ ShaderGraphContentTask(.targ...	$(VCTargetsPath)\BuildCustomizations\ShaderGraphContentTask.ta

Select masm. Click **OK**

You should be back to the original screen. Right click project name again and this time select **Properties.**



You should see:



General

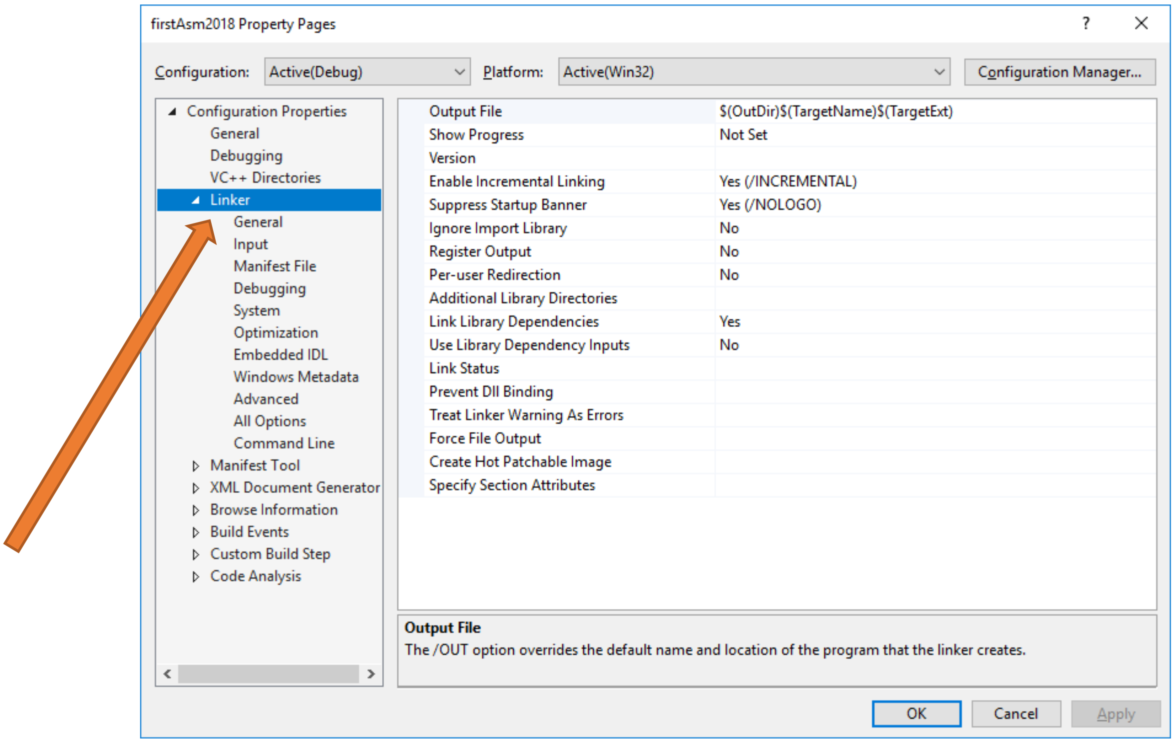
Target Platform	Windows 10
Windows SDK Version	**10.0.17134.0**
Output Directory	$(SolutionDir)$(Configuration)\
Intermediate Directory	$(Configuration)\
Target Name	$(ProjectName)
Target Extension	.exe
Extensions to Delete on Clean	*.cdf;*.cache;*.obj;*.obj.enc;*.ilk;*.ipdb;*.iobj;*.resources;*.tlb;*.tli;*.t
Build Log File	$(IntDir)$(MSBuildProjectName).log
Platform Toolset	**Visual Studio 2017 (v141)**
Enable Managed Incremental Build	No

Project Defaults

Configuration Type	**Application (.exe)**
Use of MFC	Use Standard Windows Libraries
Character Set	**Use Multi-Byte Character Set**
Common Language Runtime Support	No Common Language Runtime Support
.NET Target Framework Version	
Whole Program Optimization	No Whole Program Optimization
Windows Store App Support	No

Target Platform
The current target platform of the project.

We now need to setup the **LINKER** details. Expand the **Linker** tab.

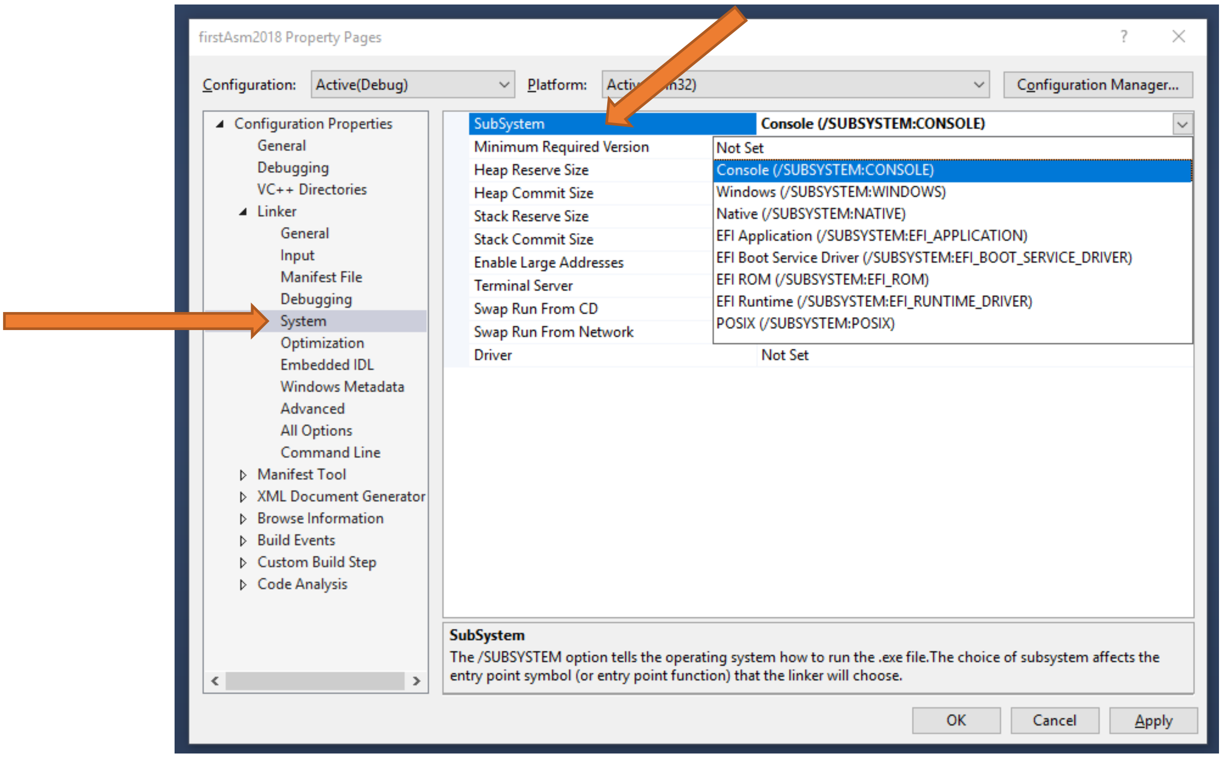


Output File	$(OutDir)$(TargetName)$(TargetExt)
Show Progress	Not Set
Version	
Enable Incremental Linking	Yes (/INCREMENTAL)
Suppress Startup Banner	Yes (/NOLOGO)
Ignore Import Library	No
Register Output	No
Per-user Redirection	No
Additional Library Directories	
Link Library Dependencies	Yes
Use Library Dependency Inputs	No
Link Status	
Prevent Dll Binding	
Treat Linker Warning As Errors	
Force File Output	
Create Hot Patchable Image	
Specify Section Attributes	

Output File
The /OUT option overrides the default name and location of the program that the linker creates.

Select the **System** tab then the **SubSystem** variable**,** you should see the various options.

Select **Console (/SUBSYSTEM:CONSOLE).** Next click **Apply** and **OK.**



firstAsm2018 Property Pages

Configuration: Active(Debug)

Platform: Active(Win32)

SubSystem: Console (/SUBSYSTEM:CONSOLE)

Minimum Required Version

Heap Reserve Size

Heap Commit Size

Stack Reserve Size

Stack Commit Size

Enable Large Addresses

Terminal Server

Swap Run From CD

Swap Run From Network

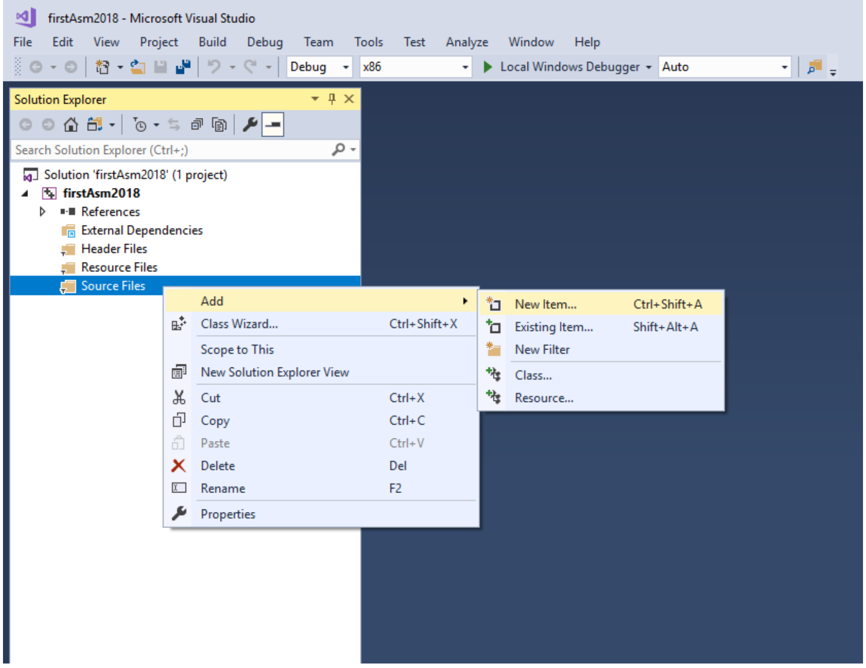
Driver: Not Set

Not Set
Console (/SUBSYSTEM:CONSOLE)
Windows (/SUBSYSTEM:WINDOWS)
Native (/SUBSYSTEM:NATIVE)
EFI Application (/SUBSYSTEM:EFI_APPLICATION)
EFI Boot Service Driver (/SUBSYSTEM:EFI_BOOT_SERVICE_DRIVER)
EFI ROM (/SUBSYSTEM:EFI_ROM)
EFI Runtime (/SUBSYSTEM:EFI_RUNTIME_DRIVER)
POSIX (/SUBSYSTEM:POSIX)

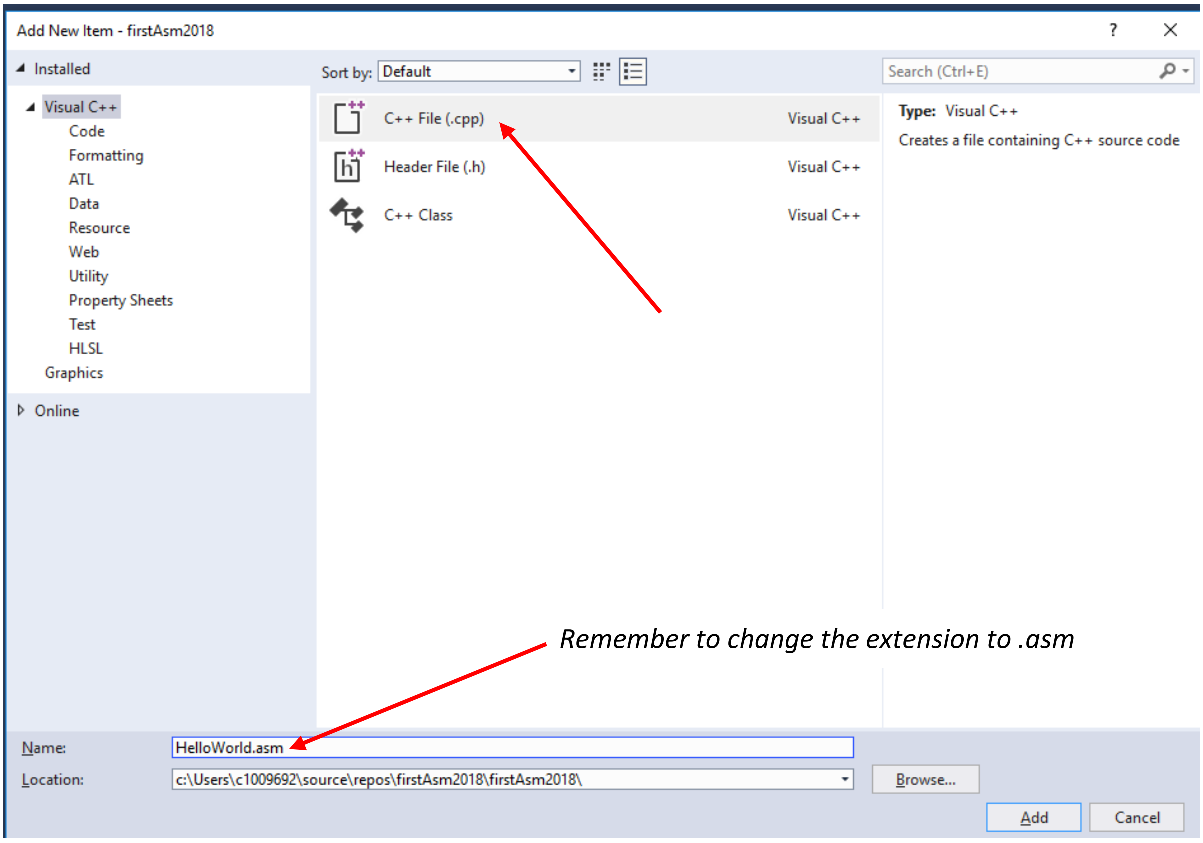
SubSystem
The /SUBSYSTEM option tells the operating system how to run the .exe file.The choice of subsystem affects the entry point symbol (or entry point function) that the linker will choose.

We have now setup our basic details for assembly. We need to add a text area so we can enter our code.

In the Solutions Explorer window right click on the source files folder select **ADD** and then **NEW ITEM**


firstAsm2018 - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug x86 Local Windows Debugger Auto
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'firstAsm2018' (1 project)
firstAsm2018
References
External Dependencies
Header Files
Resource Files
Source Files
Add
Class Wizard... Ctrl+Shift+X
Scope to This
New Solution Explorer View
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Del
Rename F2
Properties
New Item... Ctrl+Shift+A
Existing Item... Shift+Alt+A
New Filter
Class...
Resource...

You should see the following.



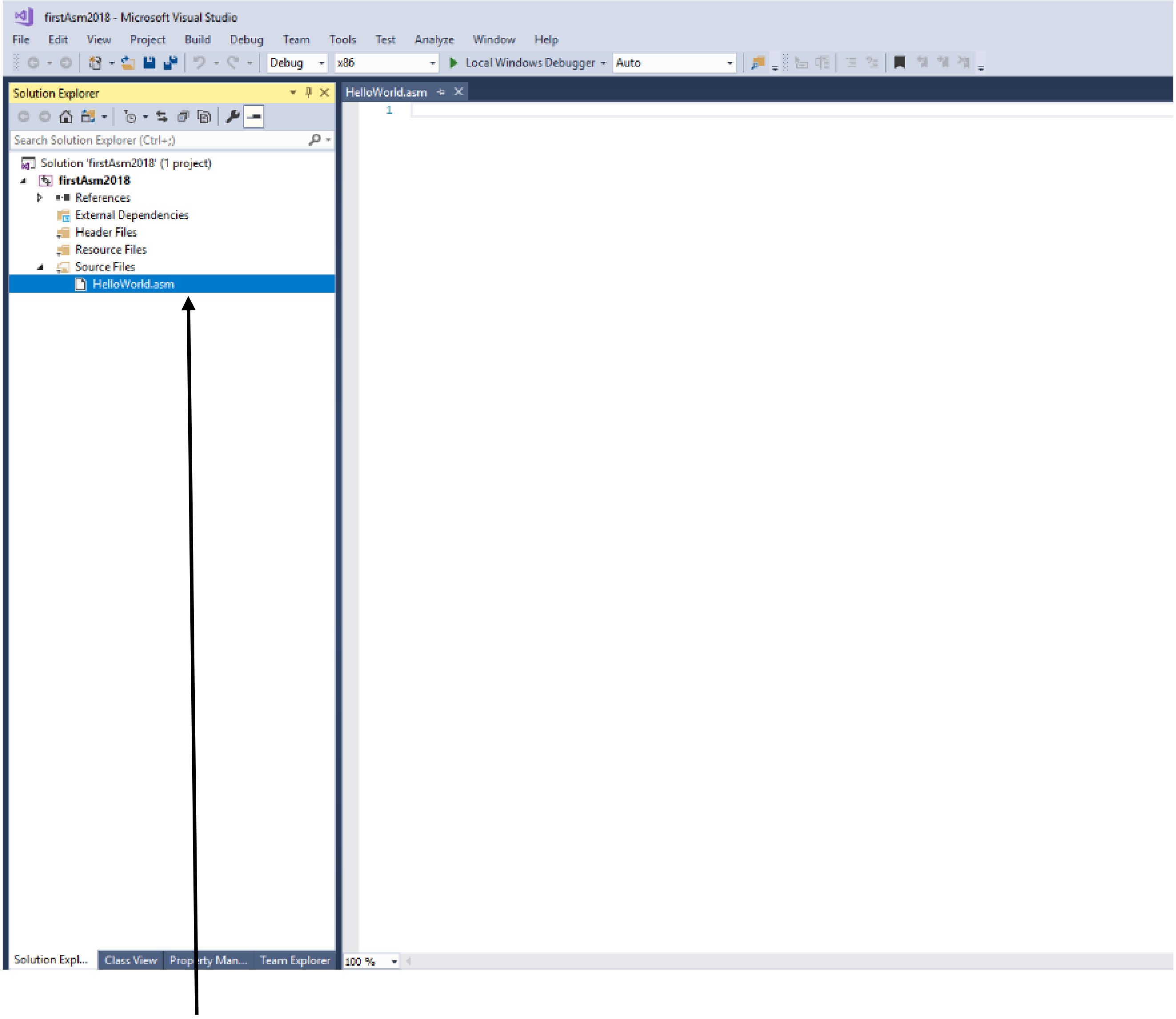
Remember to change the extension to .asm

Name: HelloWorld.asm

Location: c:\Users\c1009692\source\repos\firstAsm2018\firstAsm2018\

Select **“C++ File (.cpp)”** Type in a filename. In our case, let’s call it **HelloWorld.asm**

Click **Add.**



We now have a window called *HelloWorld.asm* to type our program in.

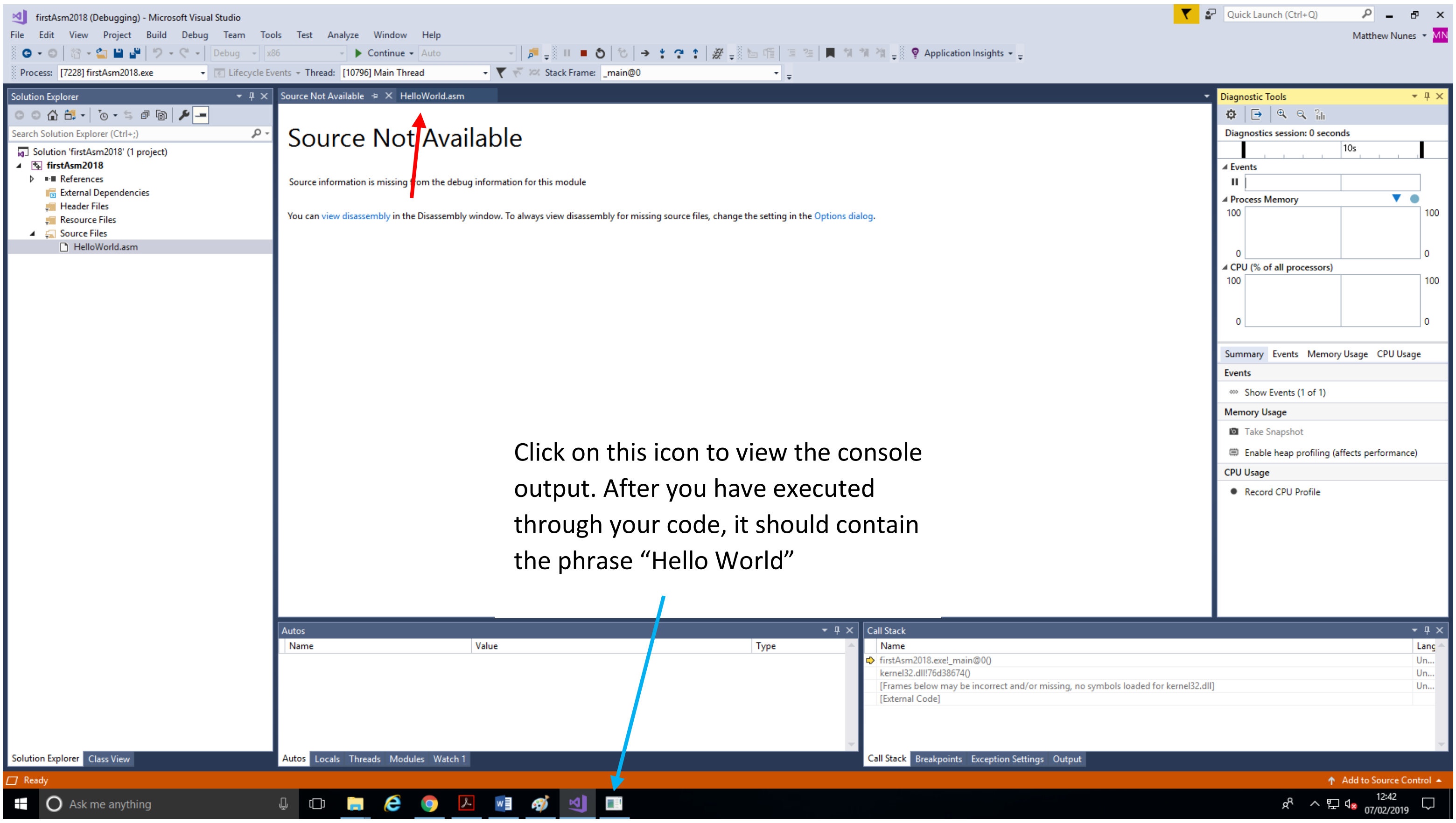
We can now enter our very first program.

TYPE in the following:

```
.586
.model flat, stdcall
option casemap :none
.stack 4096
GetStdHandle  proto :dword
ExitProcess   proto,dwExitCode:dword
WriteConsoleA        proto :dword, :dword, :dword, :dword, :dword
MessageBoxA   proto :dword, :dword, :dword, :dword

.data
       msg_txt             db     "Test program",0
       msg_caption         db     "Hello World",0
       STD_OUTPUT_HANDLE   equ    -11
       outputHandle        DWORD  ?
       bytes_written       dd     ?

.code
       main proc
       invoke GetStdHandle, STD_OUTPUT_HANDLE
       mov outputHandle, eax
       invoke WriteConsoleA,outputHandle, addr msg_caption, eax, addr bytes_written, 0
       invoke MessageBoxA, outputHandle, addr msg_txt, addr msg_caption,0

       invoke ExitProcess,0
       main endp
end main
```

Now press the **F11 function key.** If you see the screen shown below (“Source Not Available” screen), don’t worry, just click on the tab pointed to by the red arrow (called HelloWorld.asm. This is your source code). Then carry on pressing the F11 key until the end of the execution.

You should see “Hello World” in the command line at the end (it may also contain some other text, but as long as it shows “Hello World” somewhere, it’s all good):



firstAsm2018 (Debugging) - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Continue
Application Insights
Process: [7228] firstAsm2018.exe
Lifecycle Events
Thread: [10796] Main Thread
Stack Frame: _main@0
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'firstAsm2018' (1 project)
firstAsm2018
References
External Dependencies
Header Files
Resource Files
Source Files
HelloWorld.asm
Source Not Available
HelloWorld.asm
Diagnostic Tools
Diagnostics session: 0 seconds
Events
Process Memory
CPU (% of all processors)
Summary Events Memory Usage CPU Usage
Show Events (1 of 1)
Take Snapshot
Enable heap profiling (affects performance)
Record CPU Profile
Autos
Name Value Type
Call Stack
Autos Locals Threads Modules Watch 1
Call Stack Breakpoints Exception Settings Output
Ready
Add to Source Control
Ask me anything
12:42
07/02/2019

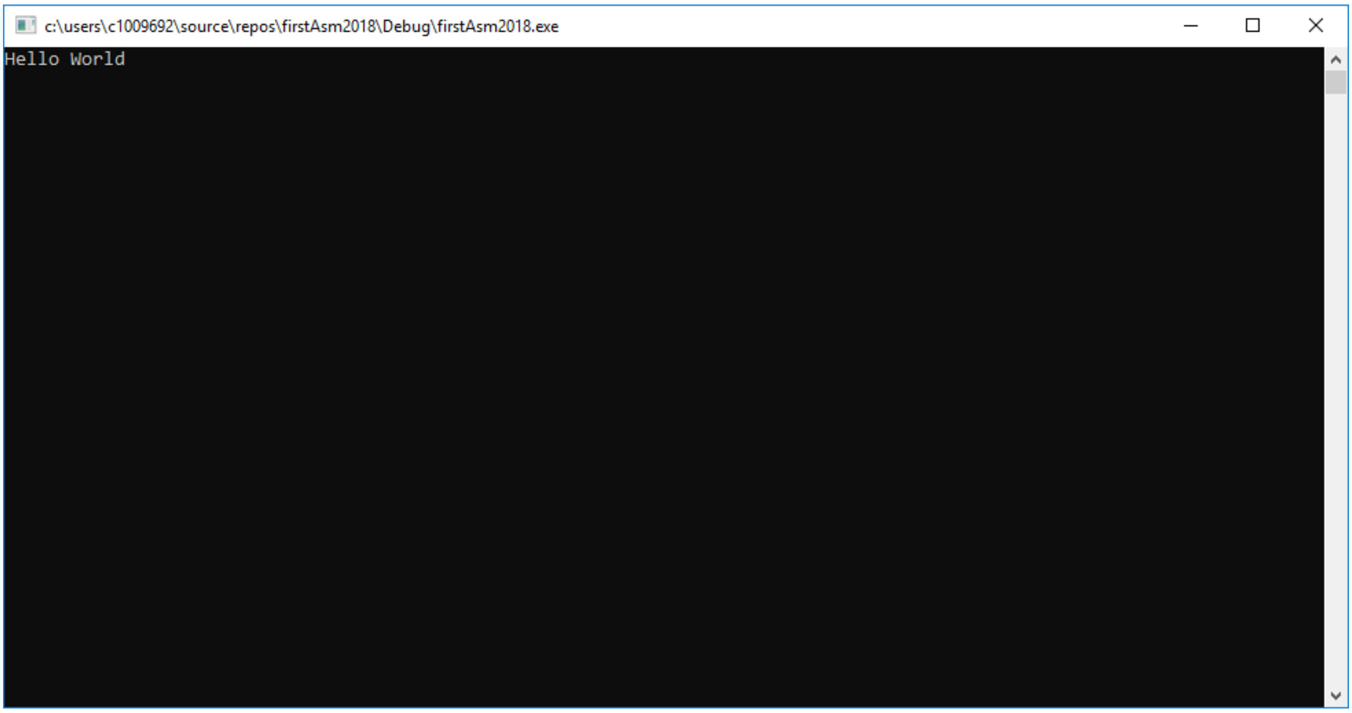
Source Not Available

Source information is missing from the debug information for this module

You can view disassembly in the Disassembly window. To always view disassembly for missing source files, change the setting in the Options dialog.

Name	Lang
firstAsm2018.exe!_main@0()	Un...
kernel32.dll!76d38674()	Un...
[Frames below may be incorrect and/or missing, no symbols loaded for kernel32.dll]	Un...
[External Code]	

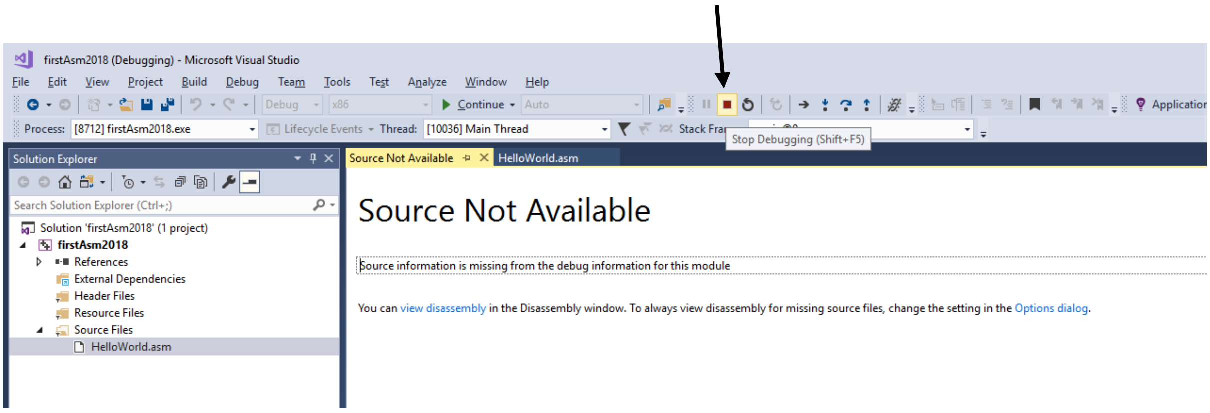
Click on this icon to view the console output. After you have executed through your code, it should contain the phrase “Hello World”



c:\users\c1009692\source\repos\firstAsm2018\Debug\firstAsm2018.exe

```
Hello World
```

If you haven’t done so already, stop execution/debugging by clicking the stop icon.

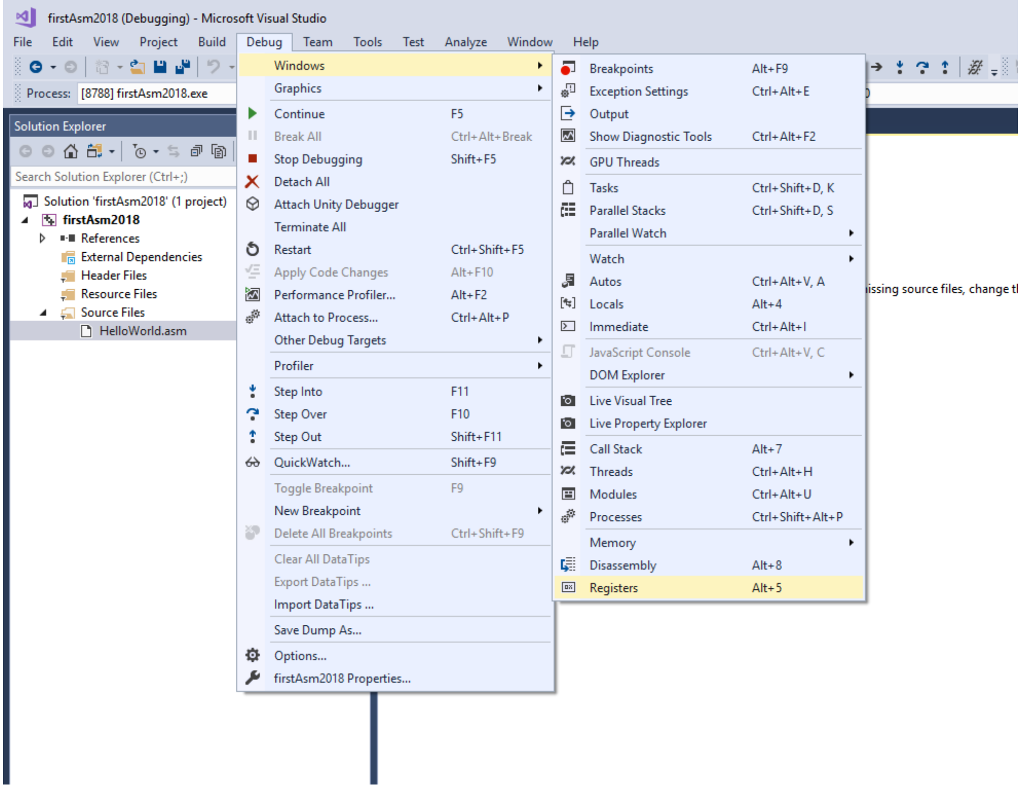

firstAsm2018 (Debugging) - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Continue
Process: [8712] firstAsm2018.exe
Lifecycle Events
Thread: [10036] Main Thread
Stack Frame
Stop Debugging (Shift+F5)
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'firstAsm2018' (1 project)
firstAsm2018
References
External Dependencies
Header Files
Resource Files
Source Files
HelloWorld.asm
Source Not Available
HelloWorld.asm

Source Not Available

Source information is missing from the debug information for this module

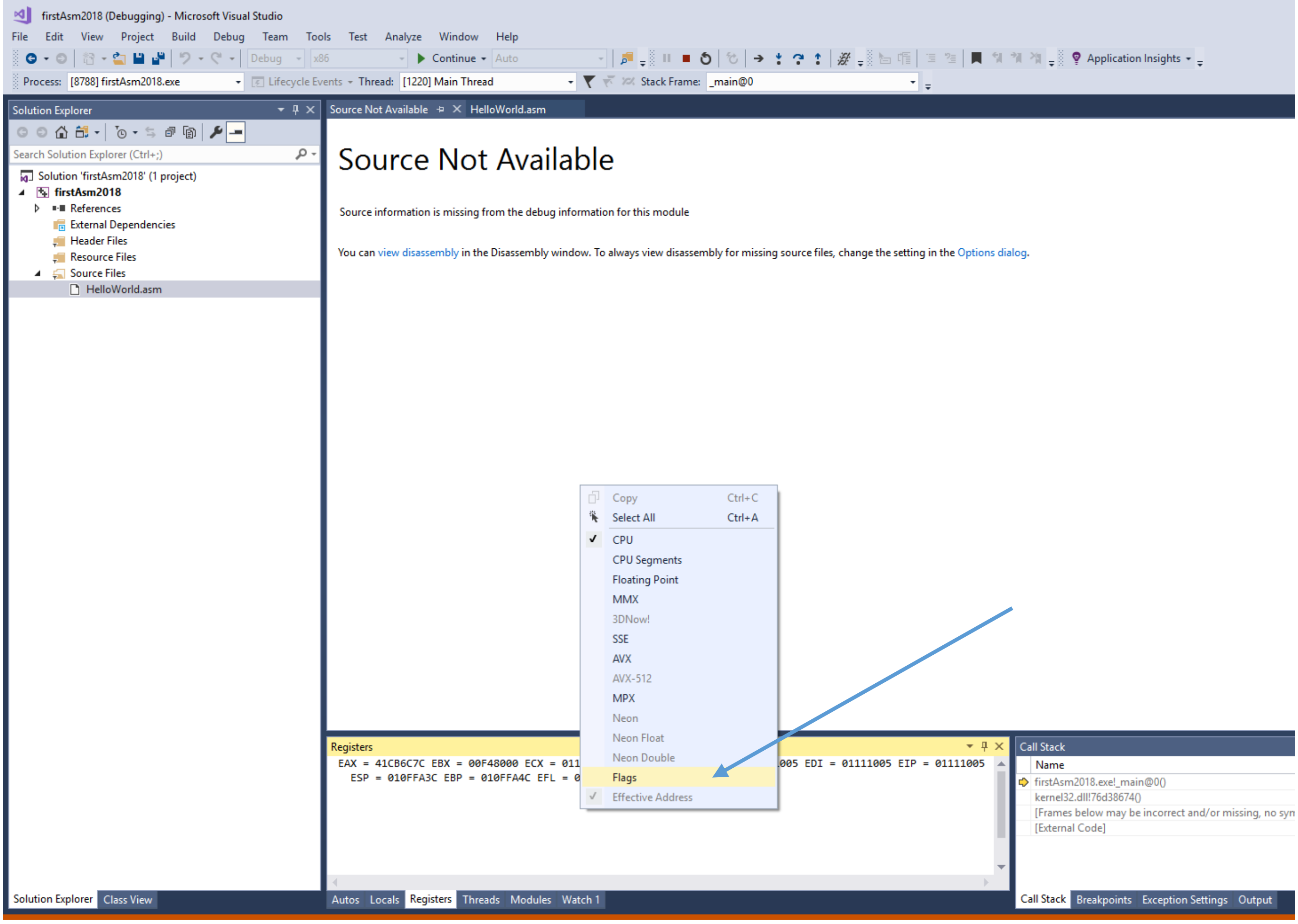
You can view disassembly in the Disassembly window. To always view disassembly for missing source files, change the setting in the Options dialog.

Let’s try going through the code again line by line, except, this time, after pressing the “**F11**” once, do the following. Click “**Debug**” in the horizontal top bar, then “**Windows**”, and finally “**Registers**”. This will let us view the register values during execution



You will now see the CPU registers displayed at the bottom (but potentially somewhere else depending on your installation).

If you right click the mouse in the window displaying the **Registers** values, you will see an option to view the **FLAGS** as well, select this option to view the Flags as well.



Source Not Available

Source information is missing from the debug information for this module

You can view disassembly in the Disassembly window. To always view disassembly for missing source files, change the setting in the Options dialog.

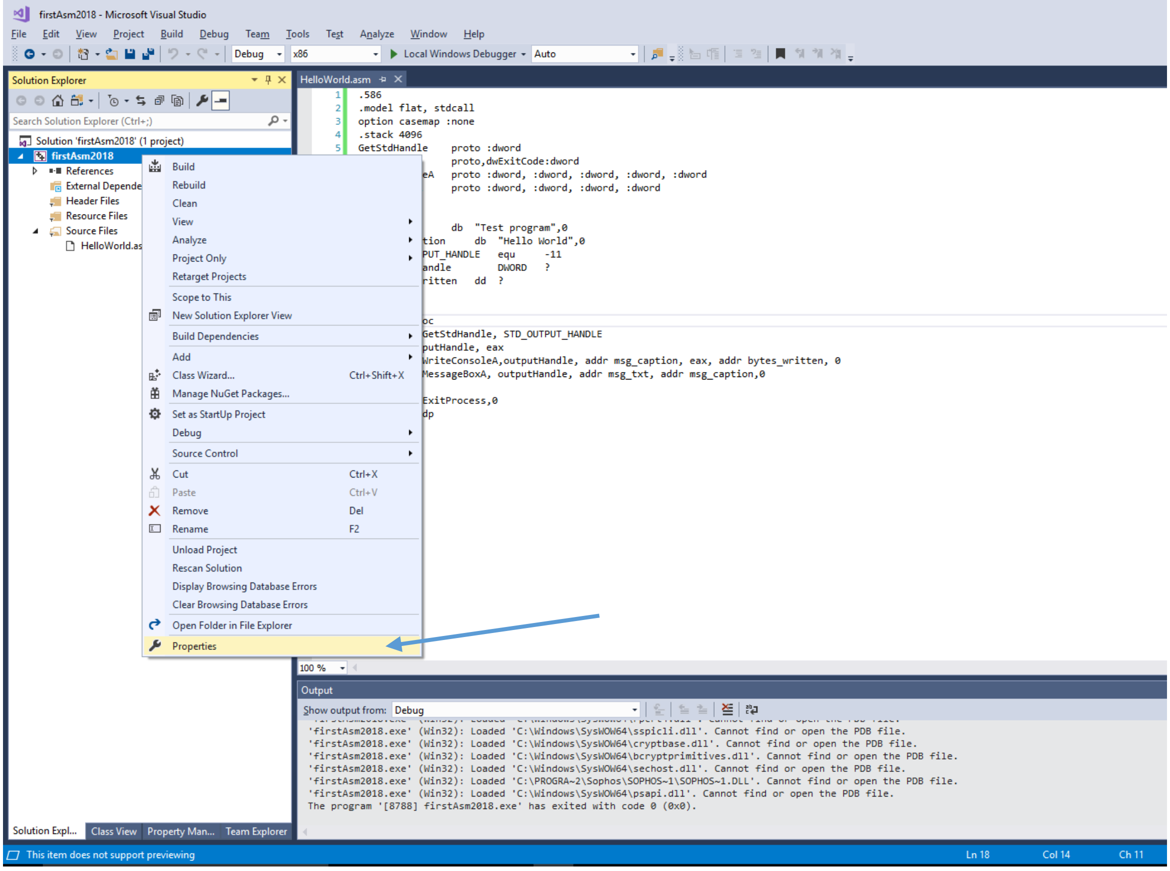
```
EAX = 41CB6C7C EBX = 00F48000 ECX = 011                          005 EDI = 01111005 EIP = 01111005
  ESP = 010FFA3C EBP = 010FFA4C EFL = 0
```

firstAsm2018.exe!_main@0()
kernel32.dll!76d38674()
[Frames below may be incorrect and/or missing, no syn
[External Code]

Try re-running your program step by step using the F11 key and watch the register values change.

OK lets have a look at what the assembler opcodes look like.

In the Solutions Explorer, right click on the **project name**, in this case, **firstAsm2018** and click **Properties**.

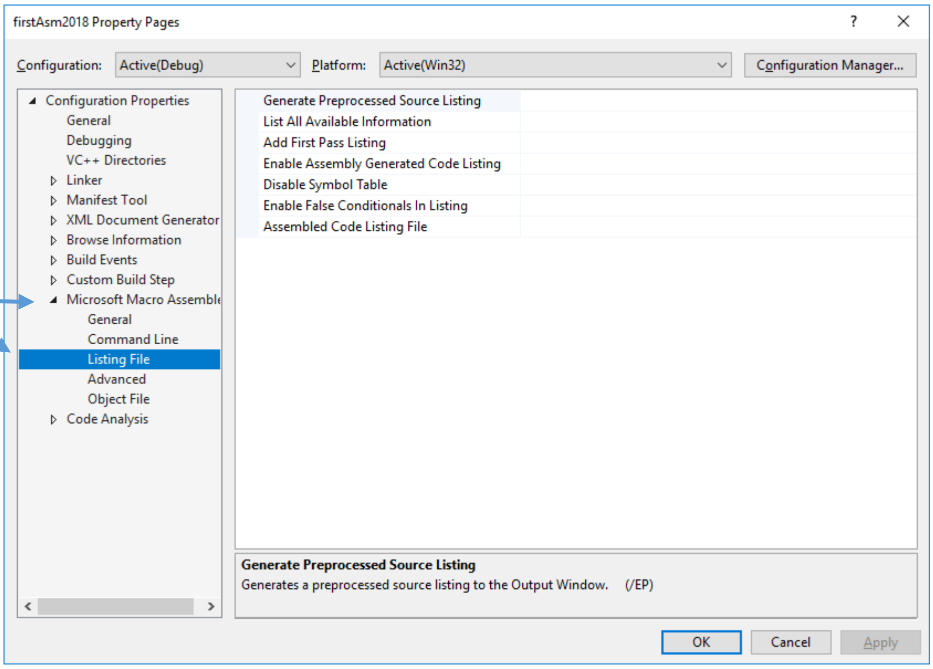

firstAsm2018 - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug x86 Local Windows Debugger Auto
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'firstAsm2018' (1 project)
firstAsm2018
References
External Depende
Header Files
Resource Files
Source Files
HelloWorld.as
Build
Rebuild
Clean
View
Analyze
Project Only
Retarget Projects
Scope to This
New Solution Explorer View
Build Dependencies
Add
Class Wizard... Ctrl+Shift+X
Manage NuGet Packages...
Set as StartUp Project
Debug
Source Control
Cut Ctrl+X
Paste Ctrl+V
Remove Del
Rename F2
Unload Project
Rescan Solution
Display Browsing Database Errors
Clear Browsing Database Errors
Open Folder in File Explorer
Properties
HelloWorld.asm
100 %
Output
Solution Expl... Class View Property Man... Team Explorer
This item does not support previewing
Ln 18 Col 14 Ch 11

```
.586
.model flat, stdcall
option casemap :none
.stack 4096
GetStdHandle    proto :dword
                proto,dwExitCode:dword
eA              proto :dword, :dword, :dword, :dword, :dword
                proto :dword, :dword, :dword, :dword


                db  "Test program",0
tion     db  "Hello World",0
PUT_HANDLE   equ     -11
andle        DWORD   ?
ritten   dd  ?


oc
GetStdHandle, STD_OUTPUT_HANDLE
putHandle, eax
WriteConsoleA,outputHandle, addr msg_caption, eax, addr bytes_written, 0
MessageBoxA, outputHandle, addr msg_txt, addr msg_caption,0

ExitProcess,0
dp
```

Show output from: Debug

```
'firstAsm2018.exe' (Win32): Loaded 'C:\Windows\SysWOW64\sspicli.dll'. Cannot find or open the PDB file.
'firstAsm2018.exe' (Win32): Loaded 'C:\Windows\SysWOW64\cryptbase.dll'. Cannot find or open the PDB file.
'firstAsm2018.exe' (Win32): Loaded 'C:\Windows\SysWOW64\bcryptprimitives.dll'. Cannot find or open the PDB file.
'firstAsm2018.exe' (Win32): Loaded 'C:\Windows\SysWOW64\sechost.dll'. Cannot find or open the PDB file.
'firstAsm2018.exe' (Win32): Loaded 'C:\PROGRA~2\Sophos\SOPHOS~1\SOPHOS~1.DLL'. Cannot find or open the PDB file.
'firstAsm2018.exe' (Win32): Loaded 'C:\Windows\SysWOW64\psapi.dll'. Cannot find or open the PDB file.
The program '[8788] firstAsm2018.exe' has exited with code 0 (0x0).
```

You should now see there is another option called **Microsoft Macro Assembler** on the lefthand side. Expand the tab. And select **Listing File**



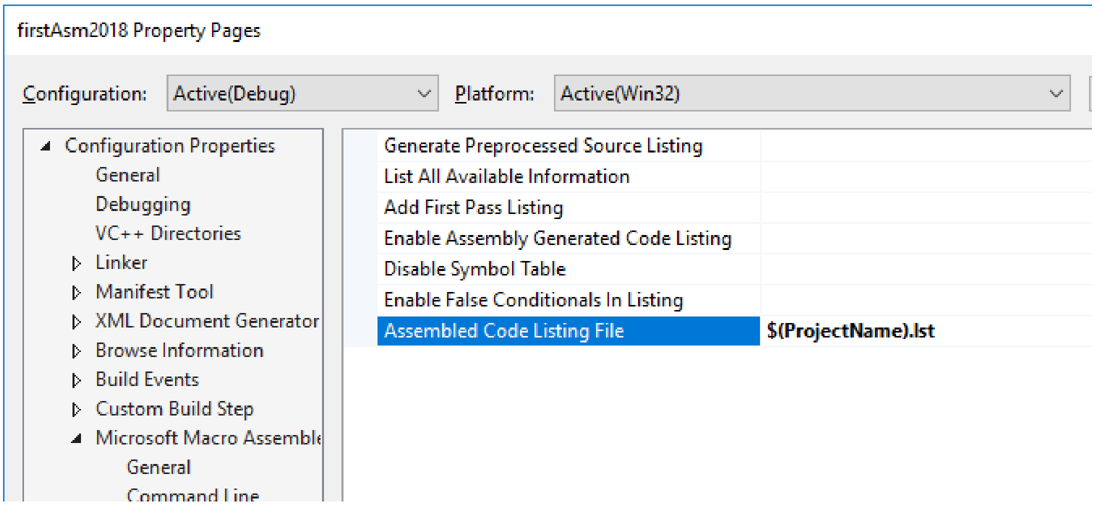
Configuration: Active(Debug)   Platform: Active(Win32)

Generate Preprocessed Source Listing
List All Available Information
Add First Pass Listing
Enable Assembly Generated Code Listing
Disable Symbol Table
Enable False Conditionals In Listing
Assembled Code Listing File

Generate Preprocessed Source Listing
Generates a preprocessed source listing to the Output Window.   (/EP)

Where you see **Assembled Code Listing file.** Add the following

$(ProjectName).lst



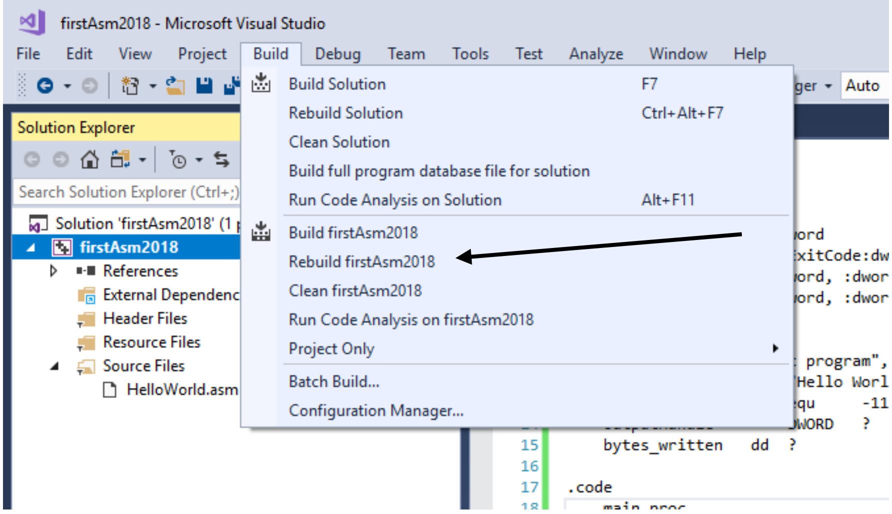
firstAsm2018 Property Pages

Configuration: Active(Debug)   Platform: Active(Win32)

Assembled Code Listing File: $(ProjectName).lst

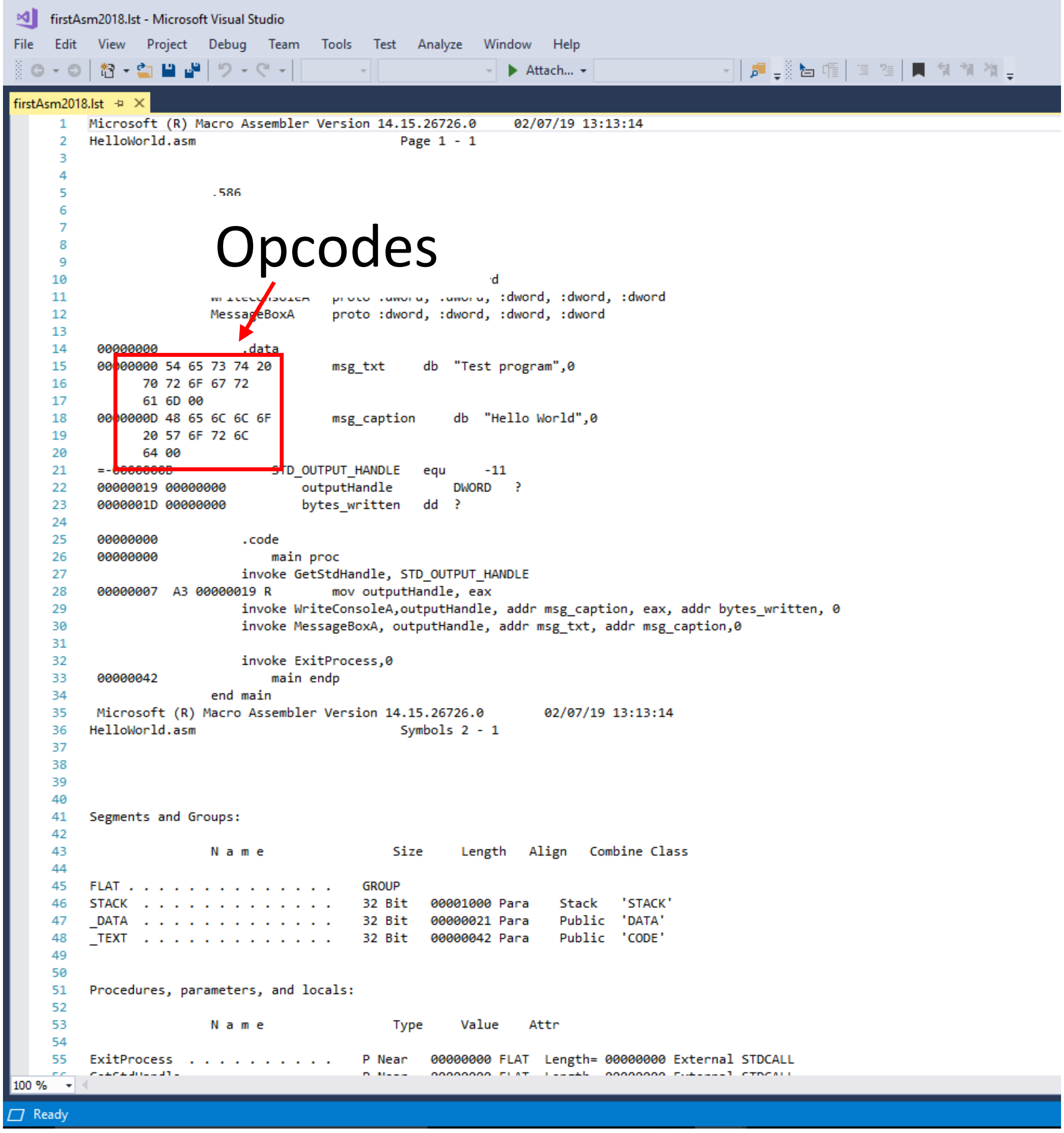
Click OK.

Now under the **Build** option select “**Rebuild firstAsm2018**”



If you look in the directory where your project is stored (*if you can’t remember where it is stored, go to “**Tools**” in the top horizontal menu, then* **Options,** *expand “**Projects and Solutions**”, and click on “**Locations**”. Your project will be stored in the directory listed under “**Projects Location**”. By default, it is C:\Users\your_username\source\repos*). In the *repos* folder, open the folder “firstAsm2018”, and the following folder with the same name (“firstAsm2018”). Here, you will see a file called ***firstAsm2018.lst***. Open the file in any Text editor (i.e. Notepad, Textpad) or double click it to open it in Visual Studio.

You should see the following.



firstAsm2018.lst - Microsoft Visual Studio
File Edit View Project Debug Team Tools Test Analyze Window Help
Attach...
firstAsm2018.lst
Opcodes
100 %
Ready

```
Microsoft (R) Macro Assembler Version 14.15.26726.0     02/07/19 13:13:14
HelloWorld.asm                                   Page 1 - 1


                .586




                                                    d
                WriteConsoleA    proto :dword, :dword, :dword, :dword, :dword
                MessageBoxA     proto :dword, :dword, :dword, :dword

 00000000                .data
 00000000 54 65 73 74 20        msg_txt     db  "Test program",0
       70 72 6F 67 72
       61 6D 00
 0000000D 48 65 6C 6C 6F        msg_caption     db  "Hello World",0
       20 57 6F 72 6C
       64 00
 =-0000000B             STD_OUTPUT_HANDLE   equ     -11
 00000019 00000000          outputHandle        DWORD   ?
 0000001D 00000000          bytes_written   dd  ?

 00000000                .code
 00000000                    main proc
                        invoke GetStdHandle, STD_OUTPUT_HANDLE
 00000007  A3 00000019 R        mov outputHandle, eax
                        invoke WriteConsoleA,outputHandle, addr msg_caption, eax, addr bytes_written, 0
                        invoke MessageBoxA, outputHandle, addr msg_txt, addr msg_caption,0

                        invoke ExitProcess,0
 00000042                    main endp
                end main
 Microsoft (R) Macro Assembler Version 14.15.26726.0        02/07/19 13:13:14
HelloWorld.asm                                   Symbols 2 - 1




Segments and Groups:

                N a m e                 Size     Length   Align   Combine Class

FLAT . . . . . . . . . . . . . .    GROUP
STACK  . . . . . . . . . . . . .    32 Bit   00001000 Para    Stack   'STACK'
_DATA  . . . . . . . . . . . . .    32 Bit   00000021 Para    Public  'DATA'
_TEXT  . . . . . . . . . . . . .    32 Bit   00000042 Para    Public  'CODE'


Procedures, parameters, and locals:

                N a m e                 Type     Value    Attr

ExitProcess  . . . . . . . . . .    P Near   00000000 FLAT  Length= 00000000 External STDCALL
```

We have now created our first Project and Assembly code program.

EXERCISE 1

Create a new project called exercise1, set the Build Customization and Linker values as previously done above (from page 7-10).

Create a new file called example1.asm **after setting the build customisation and linker values**

Enter the following code:

```
.586
.model flat ,stdcall
.stack 4096
ExitProcess  proto,dwExitCode:dword
.data
.code
main  proc
        mov eax,0
        mov ebx,0
        mov ecx,0
        mov edx,0
        mov   al,7
        mov   bh,2
        add    al,bh
        mov   eax,0
        mov   ebx,1
        sub eax,ebx
        invoke ExitProcess,0
main  endp
end  main
```

Build the project, and single step through your program making notes on the values in the various registers and flags.

mov eax,0

EAX = EBX = ECX = EDX = ESI =
EDI = EIP = ESP = EBP = EFL =

OV = UP = EI = PL = ZR = AC = PE = CY =

mov ebx,0

EAX = EBX = ECX = EDX = ESI =
EDI = EIP = ESP = EBP = EFL =

OV = UP = EI = PL = ZR = AC = PE = CY =

mov ecx,0

EAX = EBX = ECX = EDX = ESI =
EDI = EIP = ESP = EBP = EFL =

OV = UP = EI = PL = ZR = AC = PE = CY =

mov edx,0

EAX = EBX = ECX = EDX = ESI =
EDI = EIP = ESP = EBP = EFL =

OV = UP = EI = PL = ZR = AC = PE = CY =

mov al,7

EAX = EBX = ECX = EDX = ESI =
EDI = EIP = ESP = EBP = EFL =

OV = UP = EI = PL = ZR = AC = PE = CY =

mov bh,2

EAX =   EBX =   ECX =   EDX =   ESI =
EDI =   EIP =   ESP =   EBP =   EFL =

OV =   UP =   EI =   PL =   ZR =   AC =   PE =   CY =

add al,bh

EAX =   EBX =   ECX =   EDX =   ESI =
EDI =   EIP =   ESP =   EBP =   EFL =

OV =   UP =   EI =   PL =   ZR =   AC =   PE =   CY =

mov eax,0

EAX =   EBX =   ECX =   EDX =   ESI =
EDI =   EIP =   ESP =   EBP =   EFL =

OV =   UP =   EI =   PL =   ZR =   AC =   PE =   CY =

mov ebx,1

EAX =   EBX =   ECX =   EDX =   ESI =
EDI =   EIP =   ESP =   EBP =   EFL =

OV =   UP =   EI =   PL =   ZR =   AC =   PE =   CY =

sub eax,ebx

EAX =   EBX =   ECX =   EDX =   ESI =
EDI =   EIP =   ESP =   EBP =   EFL =

OV =   UP =   EI =   PL =   ZR =   AC =   PE =   CY =

EXERCISE 2

Create a new project called exercise2, set the Build Customization and Linker values as previously done above.

Create a new file called Example2.asm

Enter the following code:

```
.586p
.model flat ,stdcall
.data
.stack
.code
main PROC
        mov eax,0
        mov ebx,0
        mov ecx,0
        mov edx,0
        MOV AL,4
        DEC AL
        DEC AL
        DEC AL
        DEC AL
        DEC AL
main ENDP
END main
```

Build the project, and single step through your program until you reach the MOV AL,4 from this point onward continue to step through making notes on the state of the flags at each step, particularly the Zero Flag (which Microsoft calls **ZR** instead of **ZF** for reasons known only to Microsoft…).

MOV AL,4

OV = UP = EI = PL = ZR = AC = PE = CY =

DEC AL

OV = UP = EI = PL = ZR = AC = PE = CY =

DEC AL

OV = UP = EI = PL = ZR = AC = PE = CY =

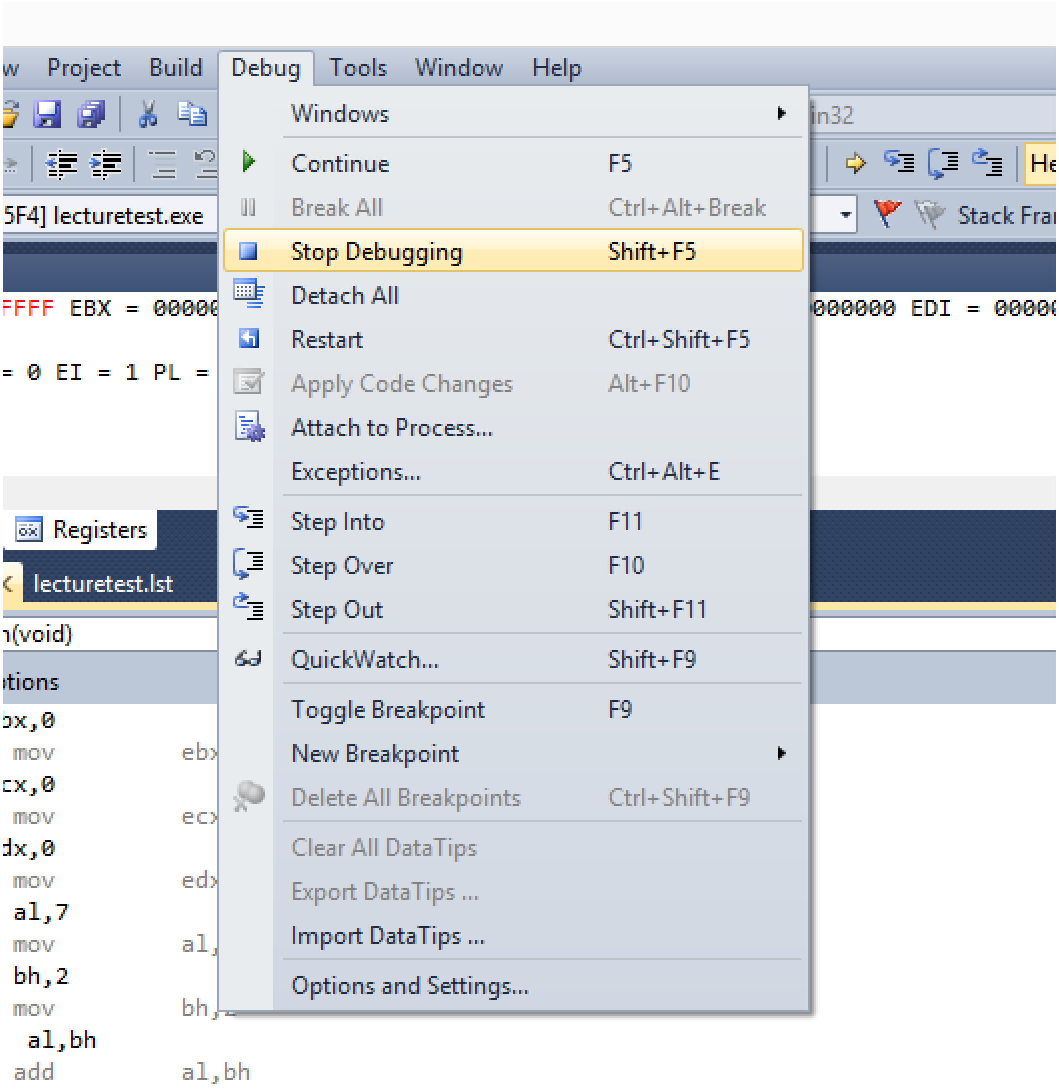
DEC AL

OV = UP = EI = PL = ZR = AC = PE = CY =

DEC AL

OV = UP = EI = PL = ZR = AC = PE = CY =

Now under DEBUG menu option STOP Debugging



This will take you back to your program. Add the following lines.

```
MOV BL, FEH
INC   BL
INC   BL
```

Re-assemble by pressing f11

Note you get an error:

error A2006: undefined symbol : FEH

REMEMBER when we are dealing in HEX if the value has a letter then we must include a leading zero
So correct your code with:

```
MOV BL,0FEH
```

Step through your previous code until you reach MOV BL,0FEH

Then record the flags as you continue.

MOV BL,0FEH

OV =   UP =   EI =   PL =   ZR =   AC =   PE =   CY =

INC BL

OV =   UP =   EI =   PL =   ZR =   AC =   PE =   CY =

INC BL

OV =   UP =   EI =   PL =   ZR =   AC =   PE =   CY =

OK That’s enough for playing with the interface. Lets try something a little more substantial

.

EXERCISE 3

Create a new project called exercise3, set the Build Customization and Linker values as previously done above.

Create a new file called Example3.asm

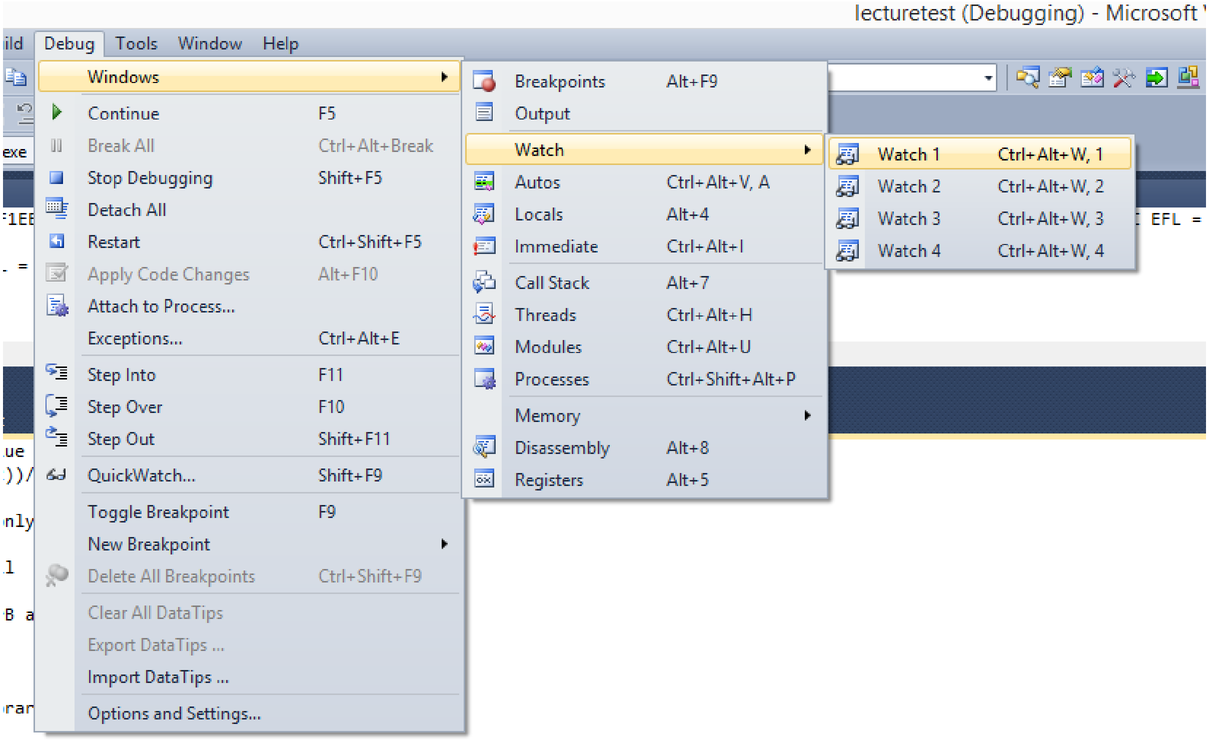
Enter the following code:

```
; Calculate the value of X, given that
; X = 3*(10A - (B+2))/(A+B) - (B+2)
;
; (This isn't the only way to do this!)
.586p
.model flat ,stdcall
.data
; Declare varA, varB and varX
       VarA   DB ?
       VarB   DB ?
       VarX   DB ?
; Some useful temporary variables
       Temp1  DB ?
       Temp2  DB ?
```

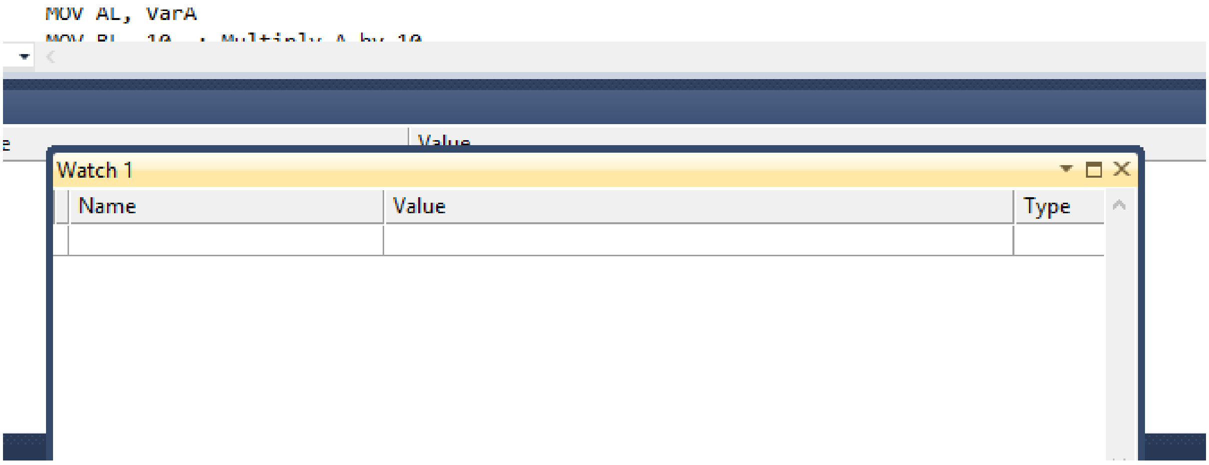
```
.stack
.code
main PROC
       MOV VarA, 4          ; Assign initial values to A
       MOV VarB, 5          ; and B (A=4, B=5)

; First calculate (A+B) and store it in Temp1
       MOV AL, VarA
       ADD AL, VarB
       MOV Temp1, AL
; Now calculate (B+2)
       MOV AL, VarB
       MOV Temp2, AL
       ADD Temp2, 2  ; Temp2 = B+2
; Now calculate 3*(10A - (B+2))
       MOV AL, VarA
       MOV BL, 10           ; Multiply A by 10
       MUL BL
       SUB AL, Temp2        ; Subtract (B+2)
       MOV BL, 3            ; Multiply the lot by 3
       MUL BL
; Can now divide by (A+B) and subtract (B+2)
       MOV BL, Temp1
       MOV AH, 0            ; extend the byte in AL into AH
       DIV BL               ; before the division
       SUB AL, Temp2        ; Subtract (B+2)
; Finally, put the result in VarX
       MOV VarX, AL
       NOP
       NOP
main ENDP
END
```

Build the project, as usual by pressing f11.

Before continuing we currently have no way of outputting our results. So we need to be able to **watch** our variables change. Go to DEBUG-WINDOWS-WATCH-WATCH1

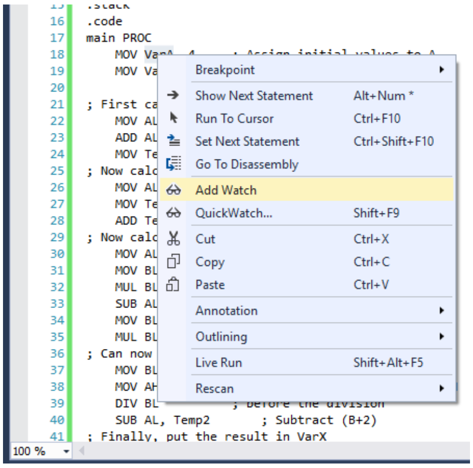


You will see a window at the bottom of the IDE



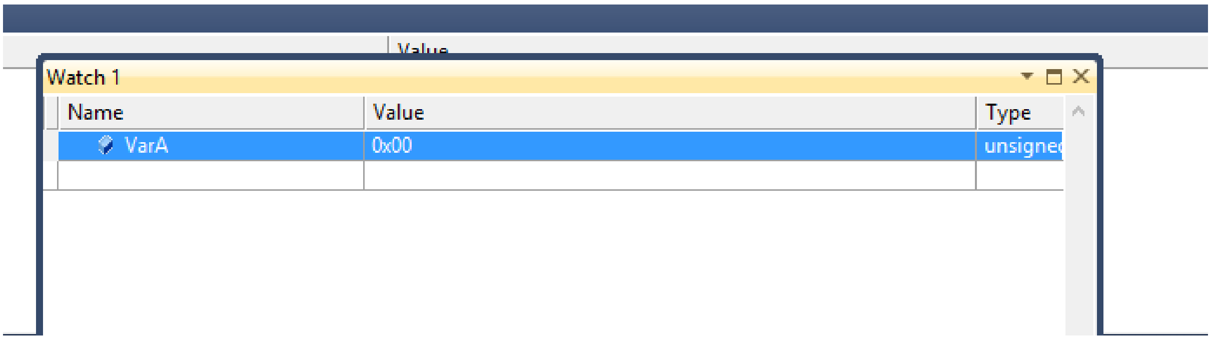
```
MOV AL, VarA
```

In your code highlight your variable (e.g varA) and right click on it and select add watch



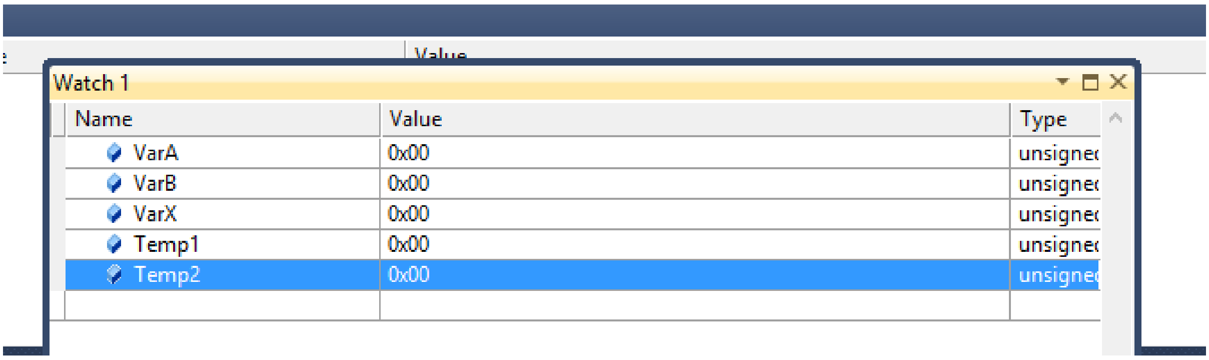
```
.code
main PROC
    MOV VarA, 4        ; Assign initial values to A
    MOV Va
; First ca
    MOV AL
    ADD AL
    MOV Te
; Now calc
    MOV AL
    MOV Te
    ADD Te
; Now calc
    MOV AL
    MOV BL
    MUL BL
    SUB AL
    MOV BL
    MUL BL
; Can now
    MOV BL
    MOV AH
    DIV BL              ; before the division
    SUB AL, Temp2       ; Subtract (B+2)
; Finally, put the result in VarX
```

You should see the varA added to the watch window.



Name	Value	Type
VarA	0x00	unsigned

Do this for each variable.



Name	Value	Type
VarA	0x00	unsigned
VarB	0x00	unsigned
VarX	0x00	unsigned
Temp1	0x00	unsigned
Temp2	0x00	unsigned

Now step through your program and write answer here

Ans: