Hands-on Learning Exercises to Accompany Chapter 4:

Assembling With VS 2008

Here is a basic helloworld.asm program in x86 assembly.

```
.386
.model flat, stdcall
option casemap :none

extrn MessageBoxA@16 : PROC
extrn ExitProcess@4 : PROC

.data
    HelloWorld db "Hello There!",0

.code
start:

    lea  eax, HelloWorld
    mov  ebx, 0
    push ebx
    push eax
    push eax
    push ebx
    call MessageBoxA@16
    push ebx
    call ExitProcess@4

end start
```

Copy and paste this a text editor and name the file helloworld.asm

There are 3 main parts to an assembly program:

1) Data

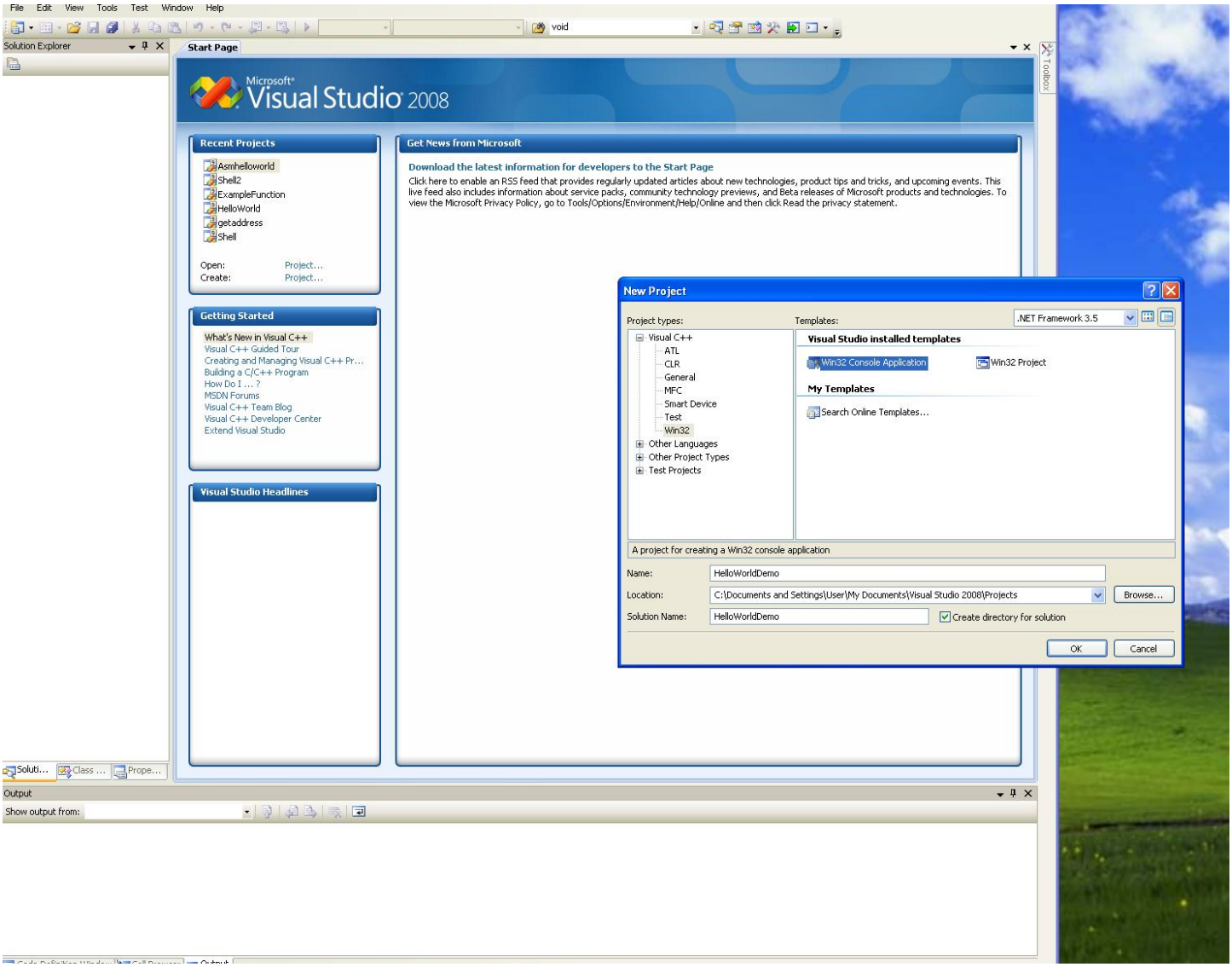
   Example: Helloworld db “Hello There!”,0

2) Code – These consist of opcodes or instructions
   Example lea eax, HelloWorld
   Push ebx

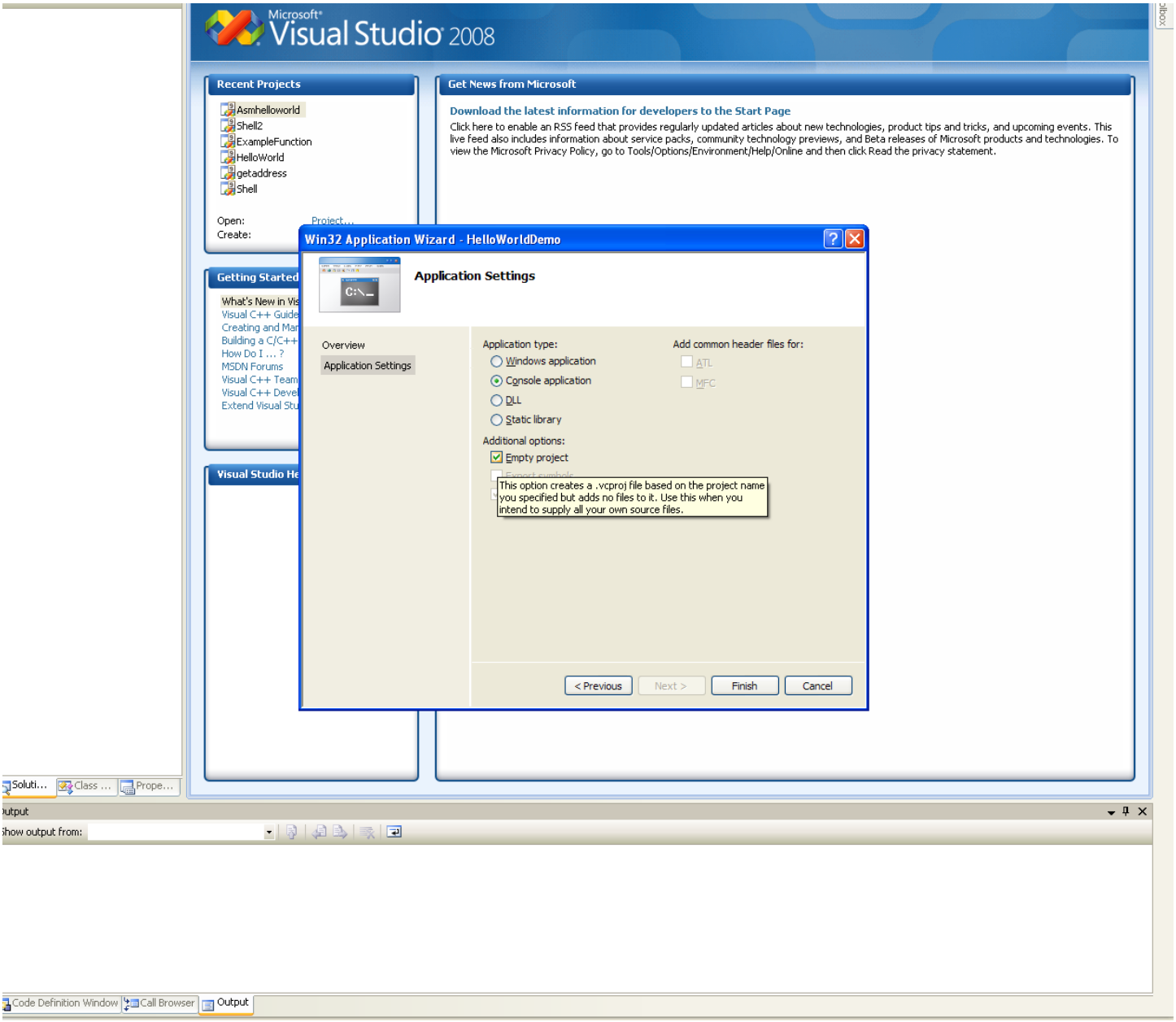
3) Directives – These tell the assembler preprocessor how to interpret and assemble. These are not considered part of the assembly code.
   Extrn
   .386

We will be using the MASM assembler as part of Visual Studio.

To assemble and run this, in the VM, open Visual Studio 2008. Create new Project,

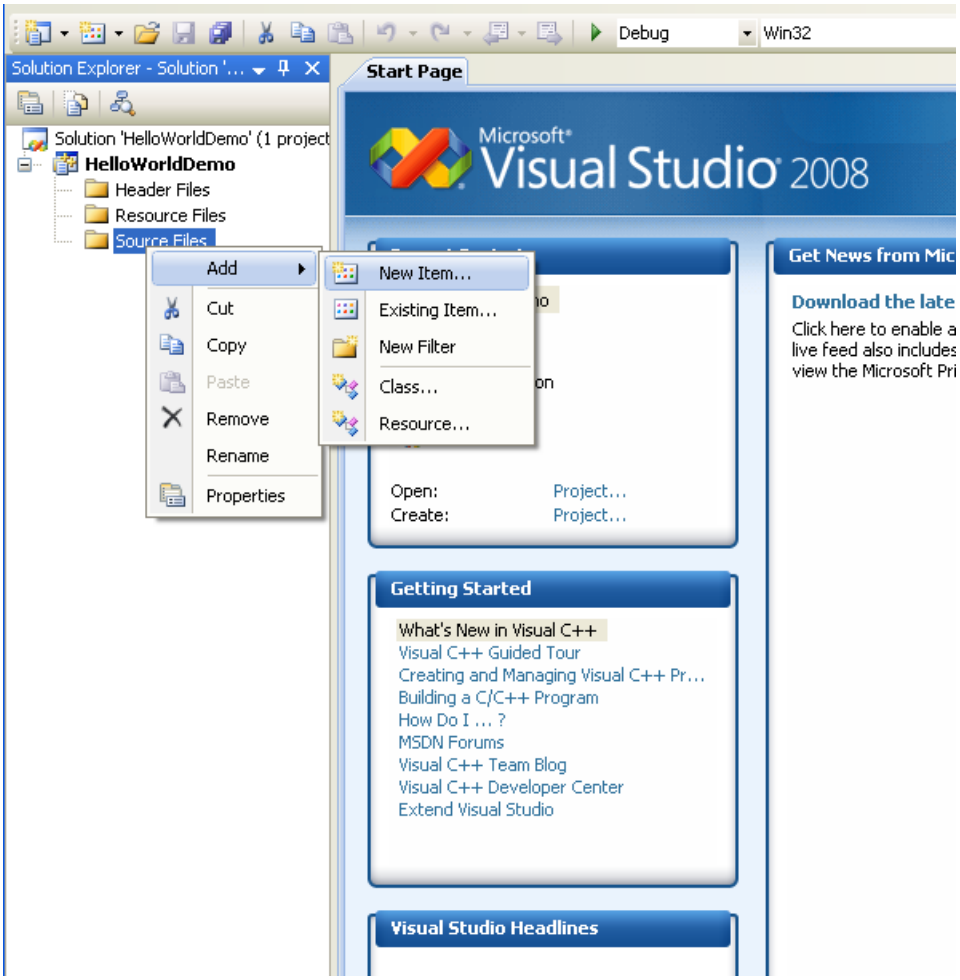


Click OK, then next, then check the box that says Empty Project

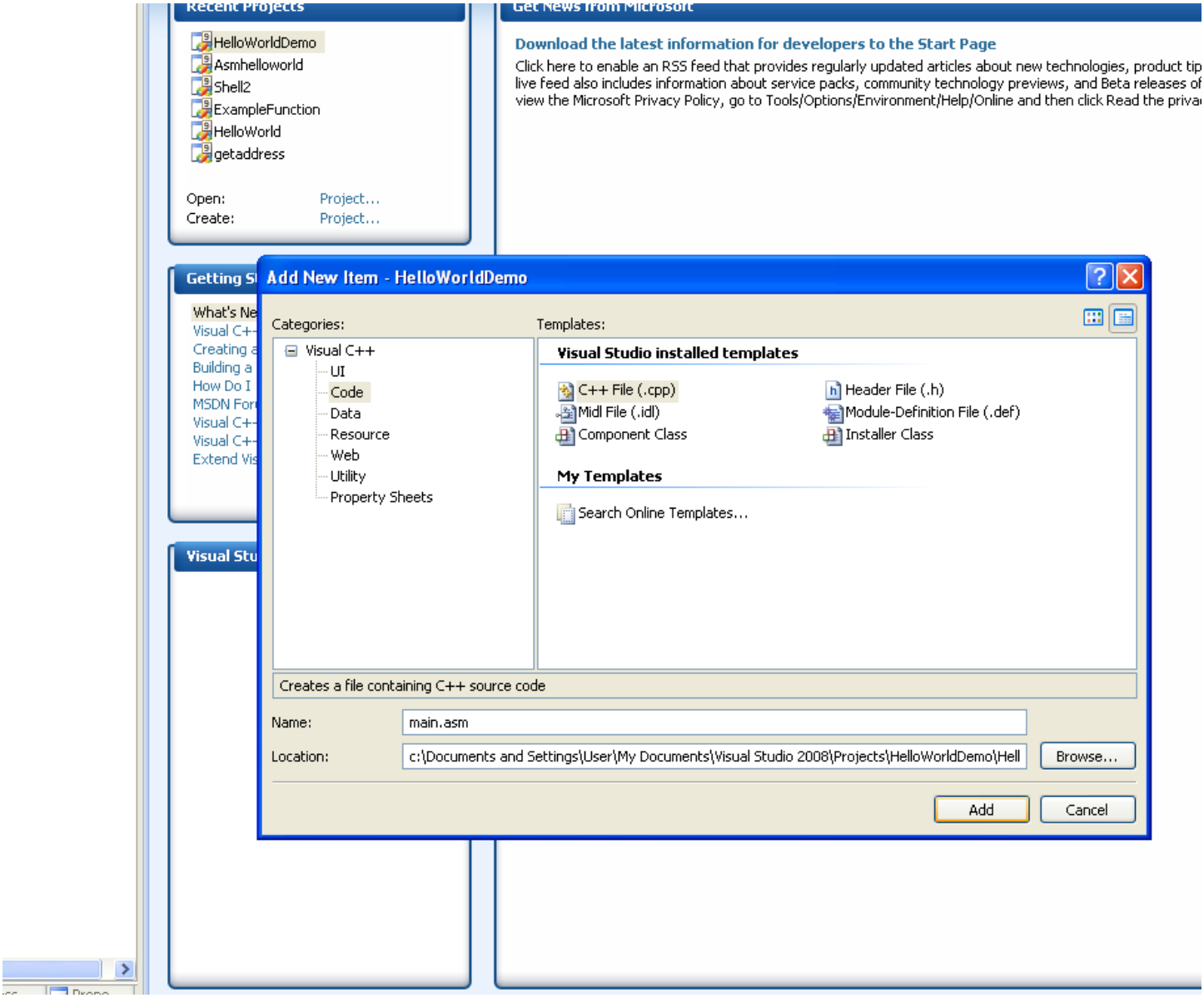


Then click Finish.

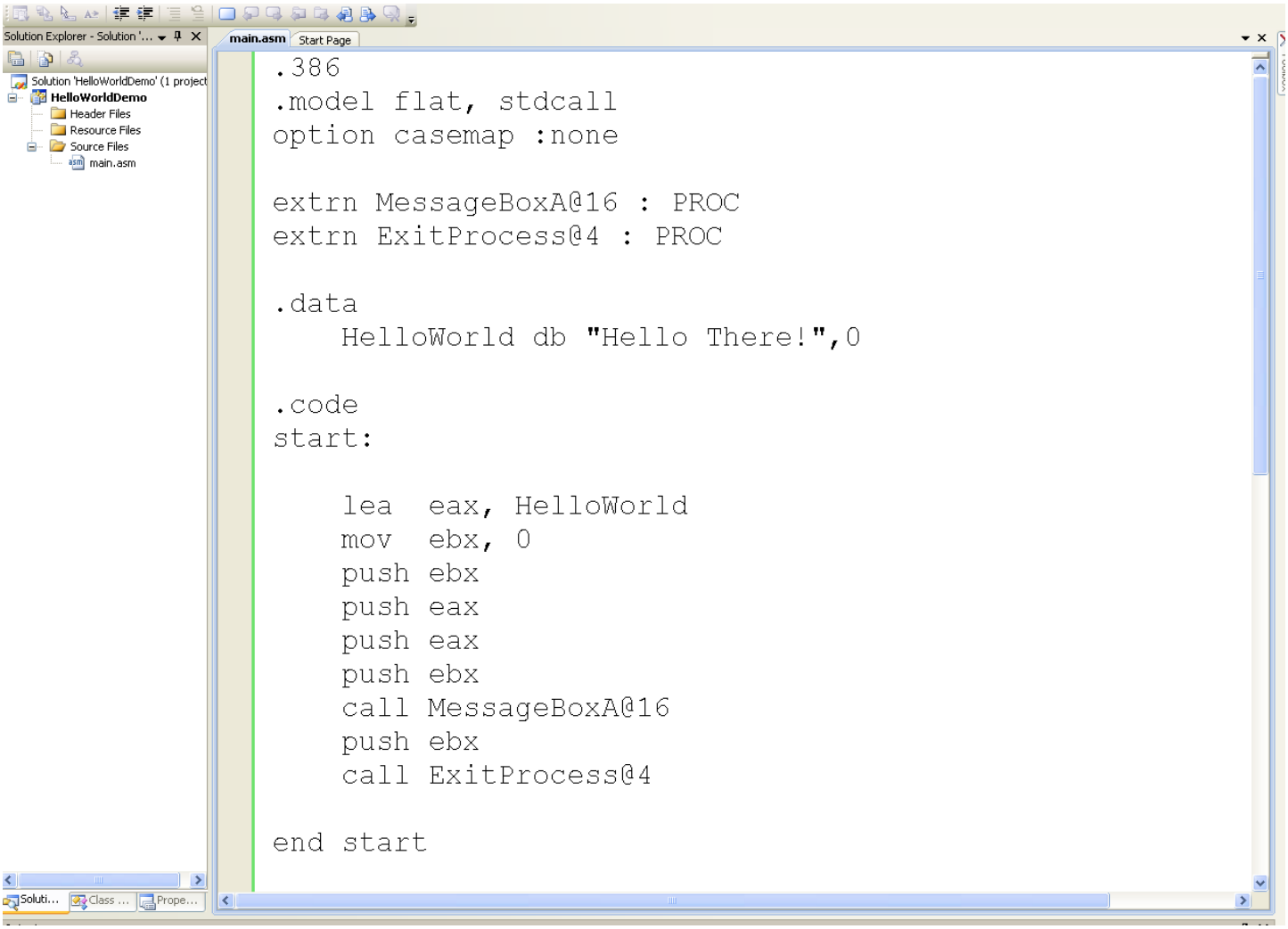
Next right-click the sources folder and go to add new item



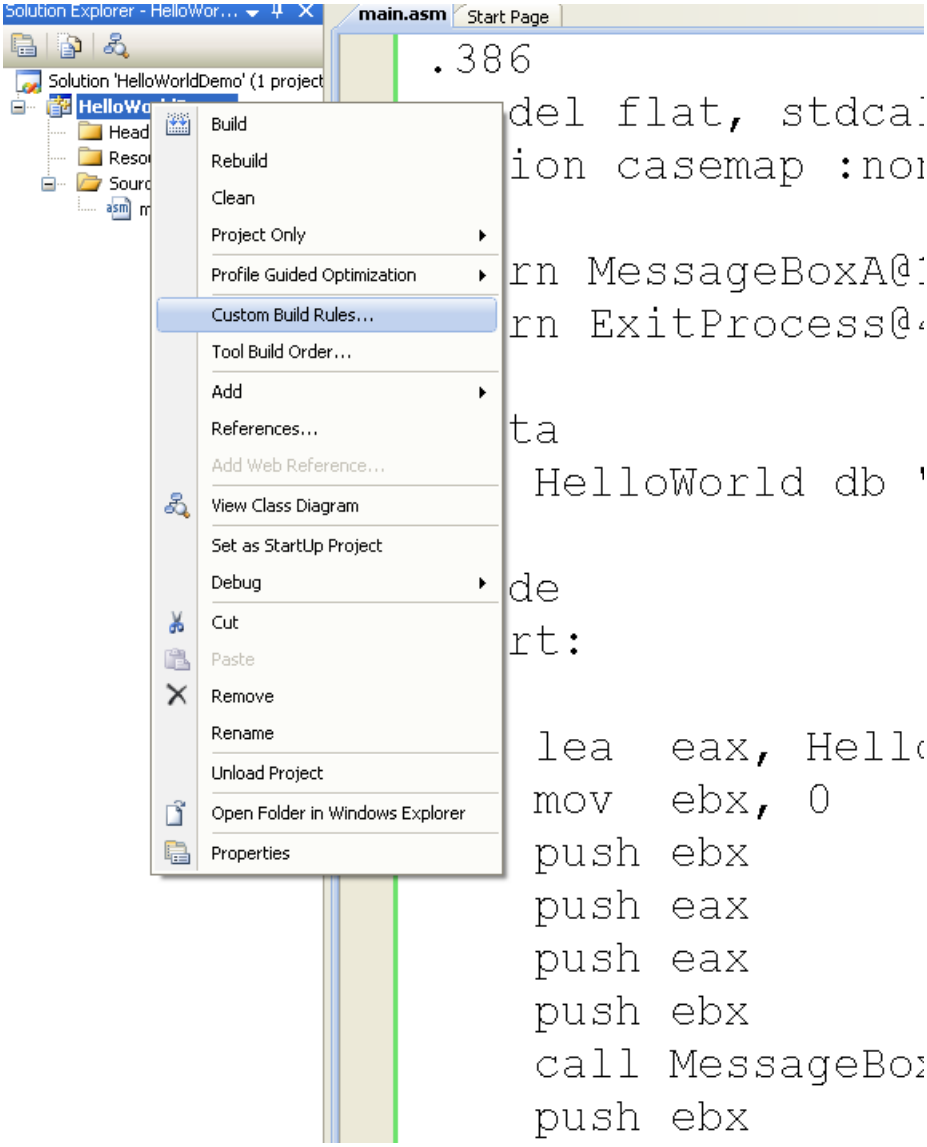
Name the new item main.asm



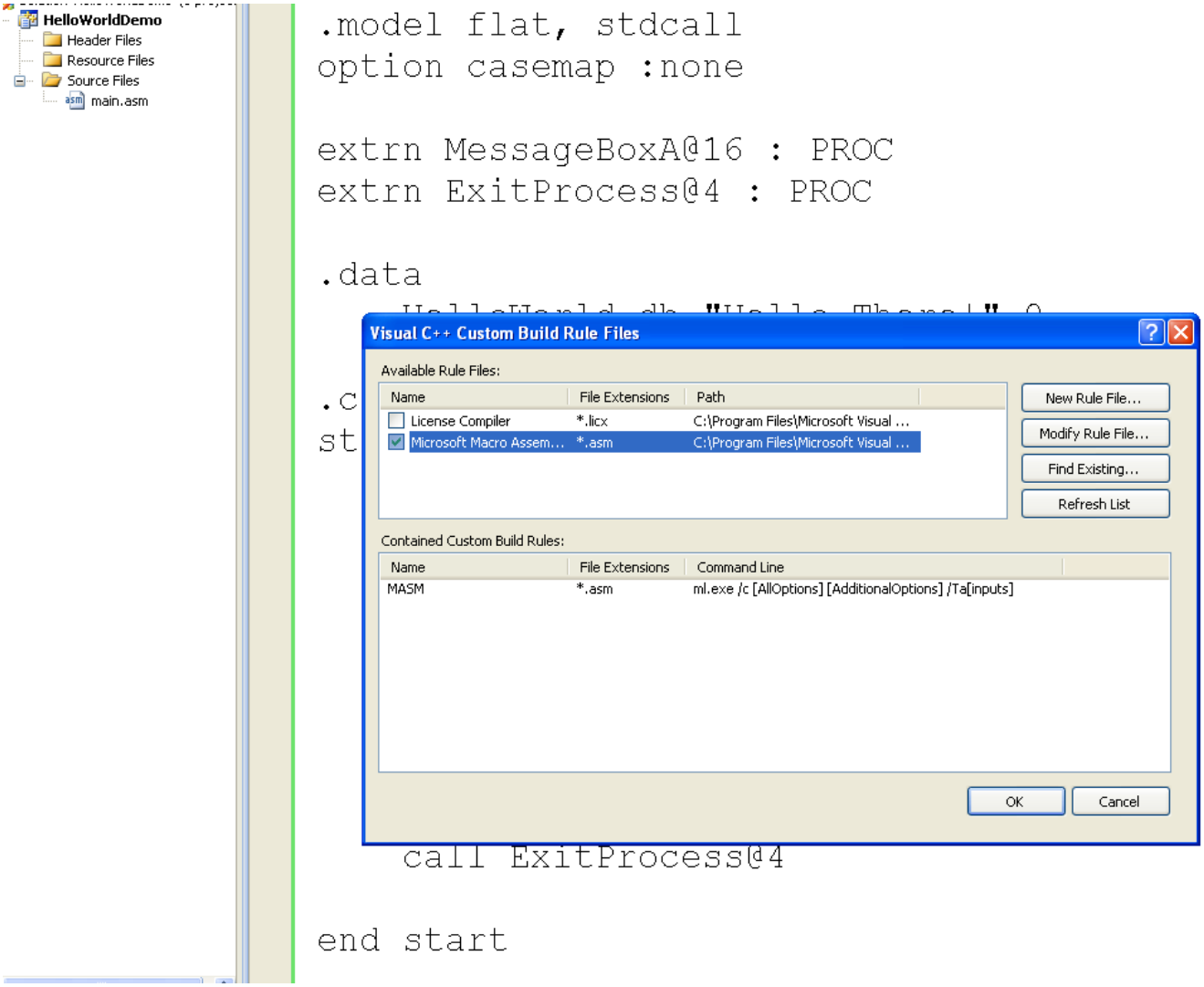
Copy and paste the assembly from the top of this document:



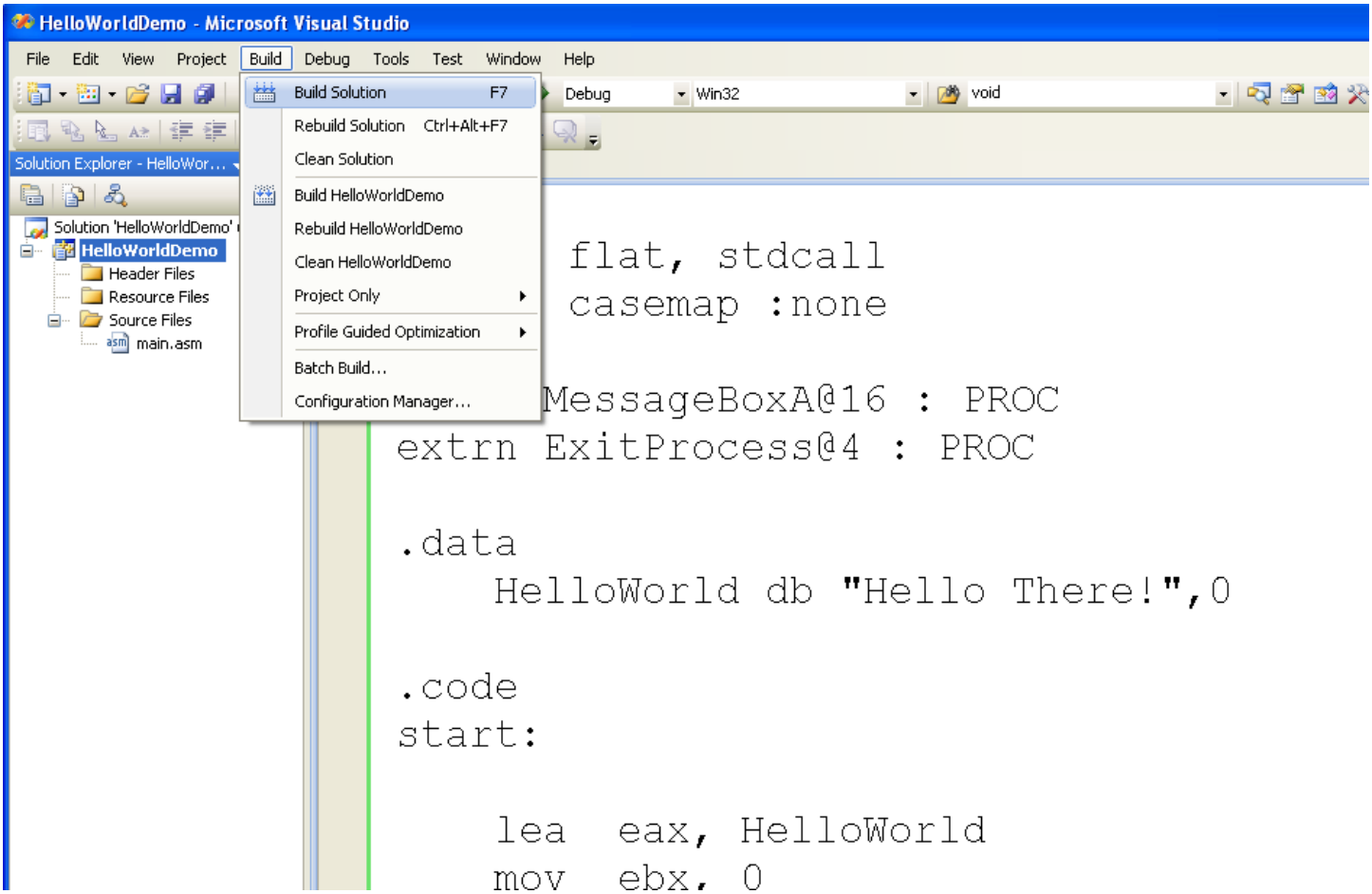
Right-click on the project and go to Custom Build Rules:



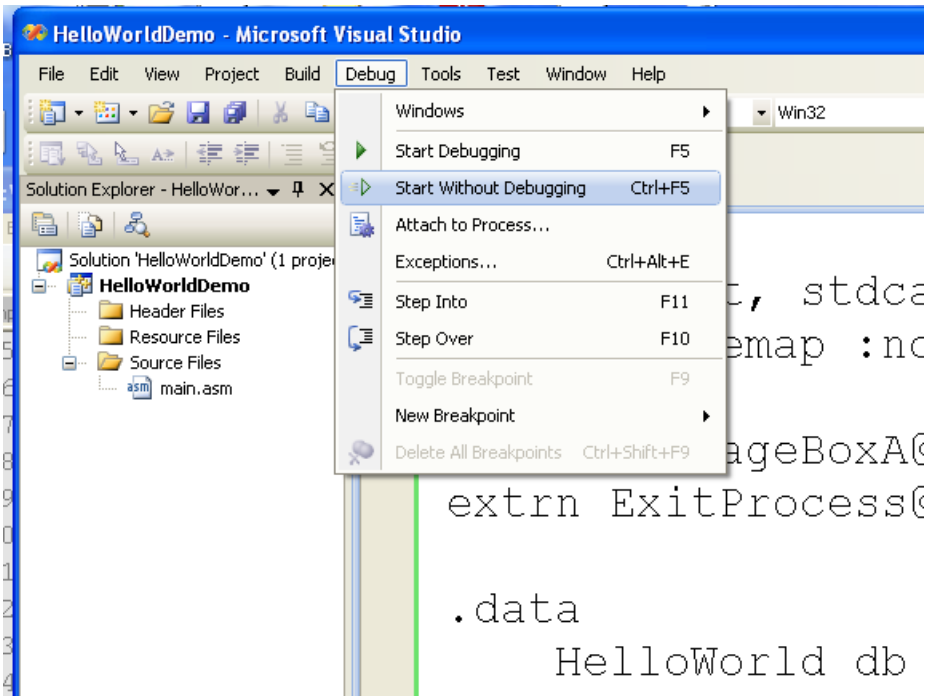
Then check the box marked Microsoft Macro Assembler.



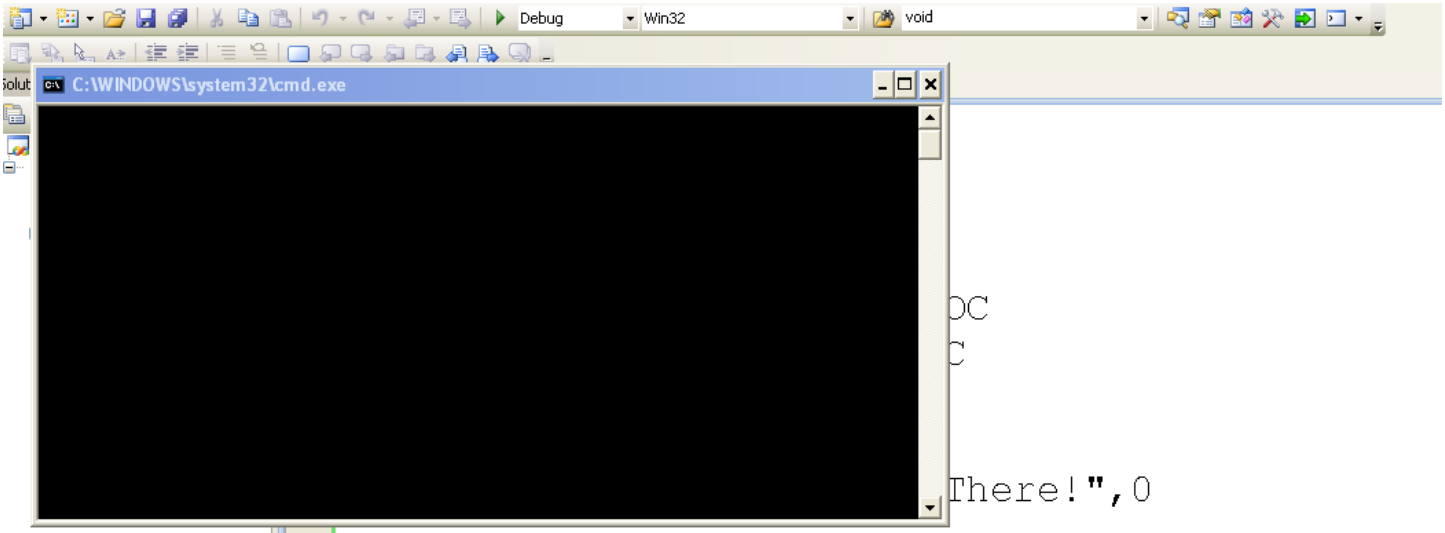
Then go to the top toolbar: Build→Build Solution



Then the top toolbar: Debug→Start Without Debugging



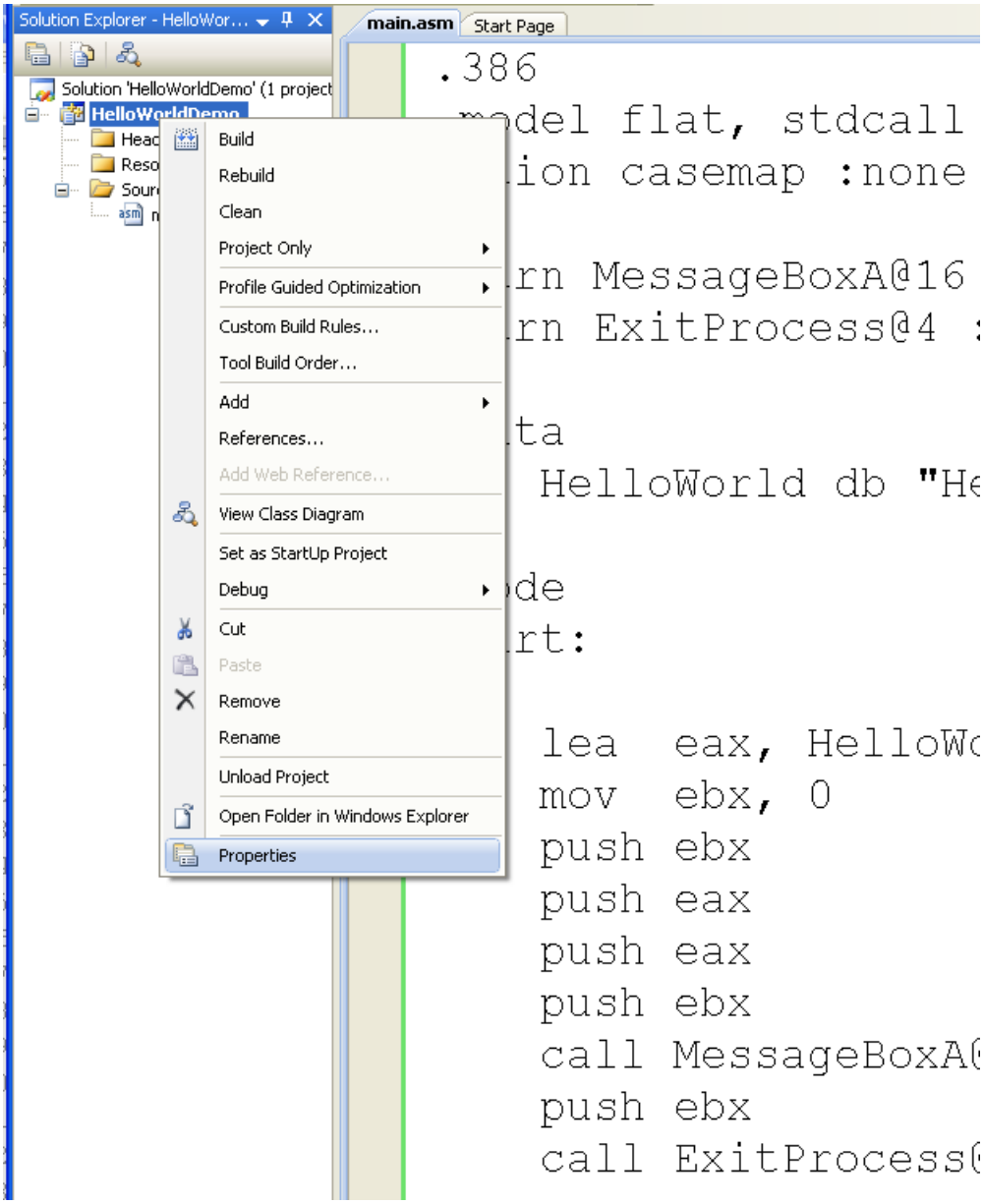
You have no successfully assembled and ran a win32 assembly program.



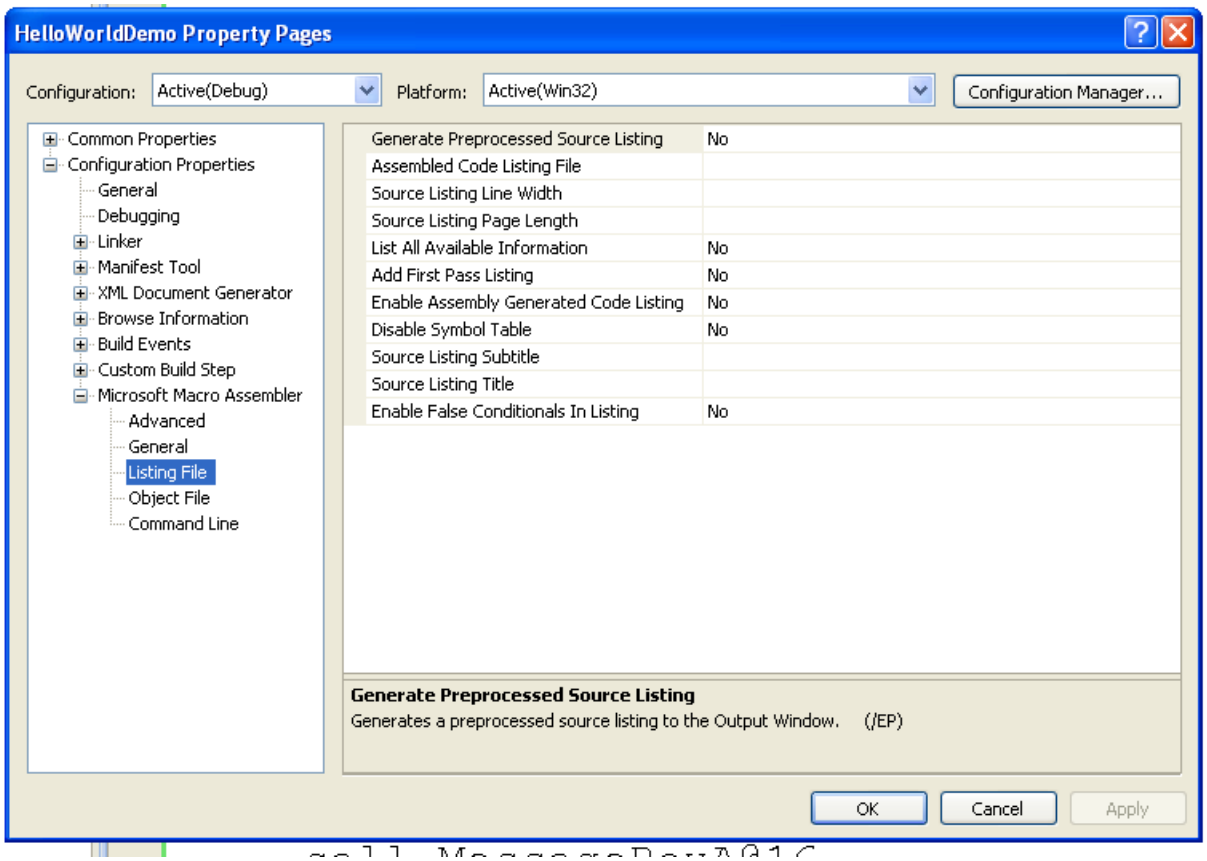

```
OC
C


There!",0

.code
start:

    lea  eax, HelloWorld
    mov  ebx, 0
    push ebx
    push eax
    push eax
    push ebx
    call MessageBoxA@16
    push ebx
    call ExitProcess@4
```

Now we will look at what those instructions are translated into.

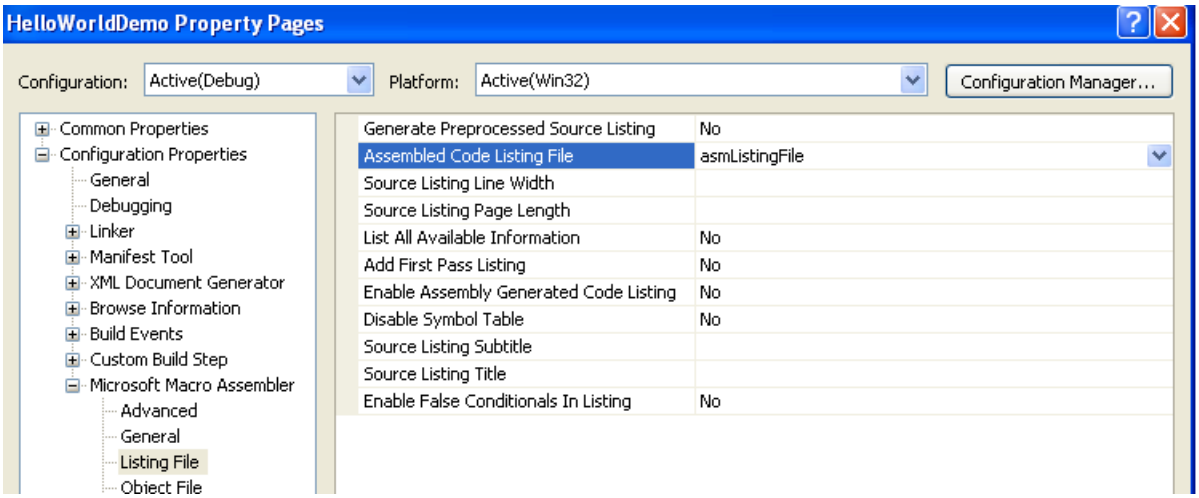
Right-click the project again and go to properties.



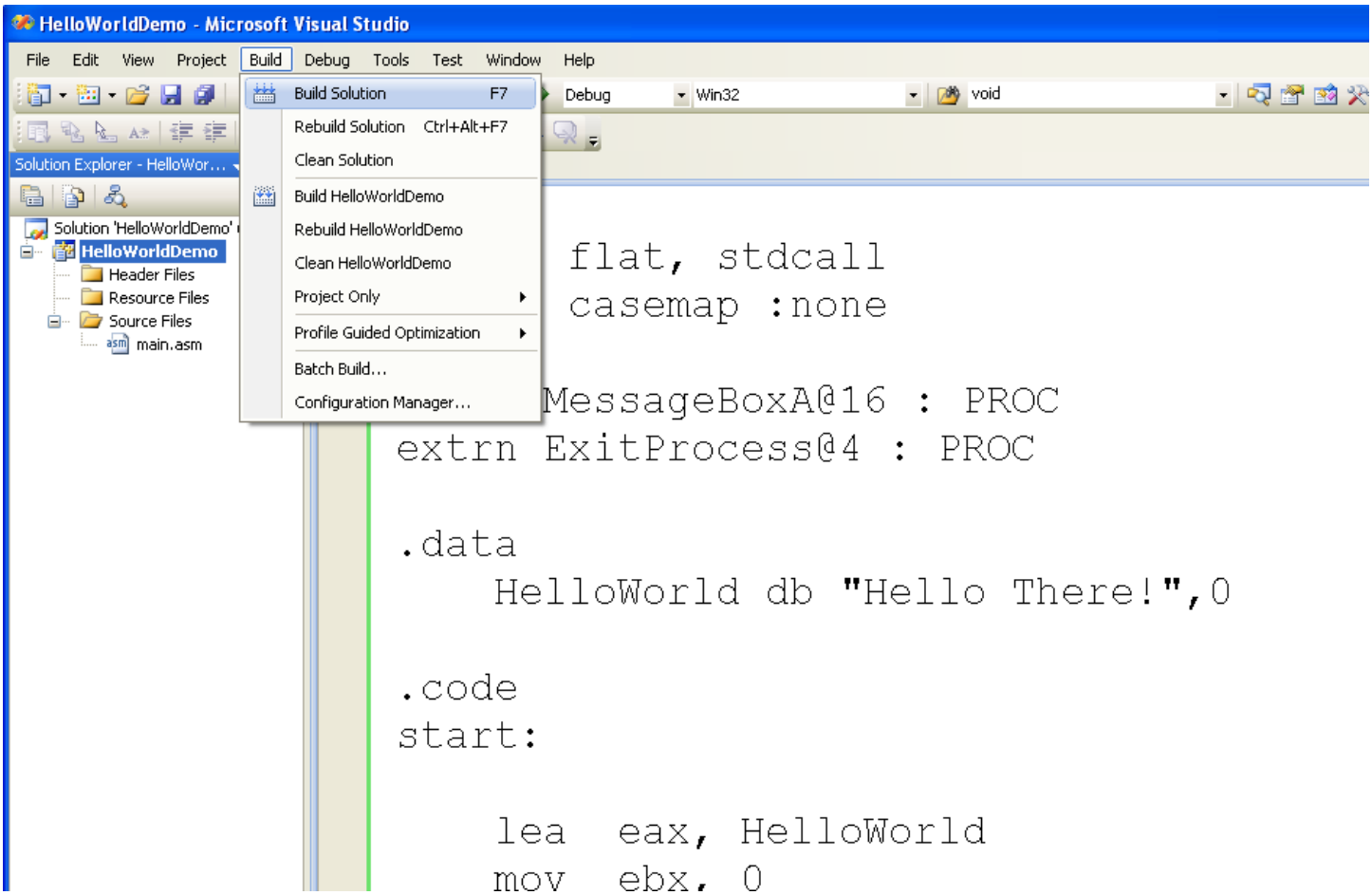
Go to Configuration Properties→Microsoft Macro Assembler→Listing File



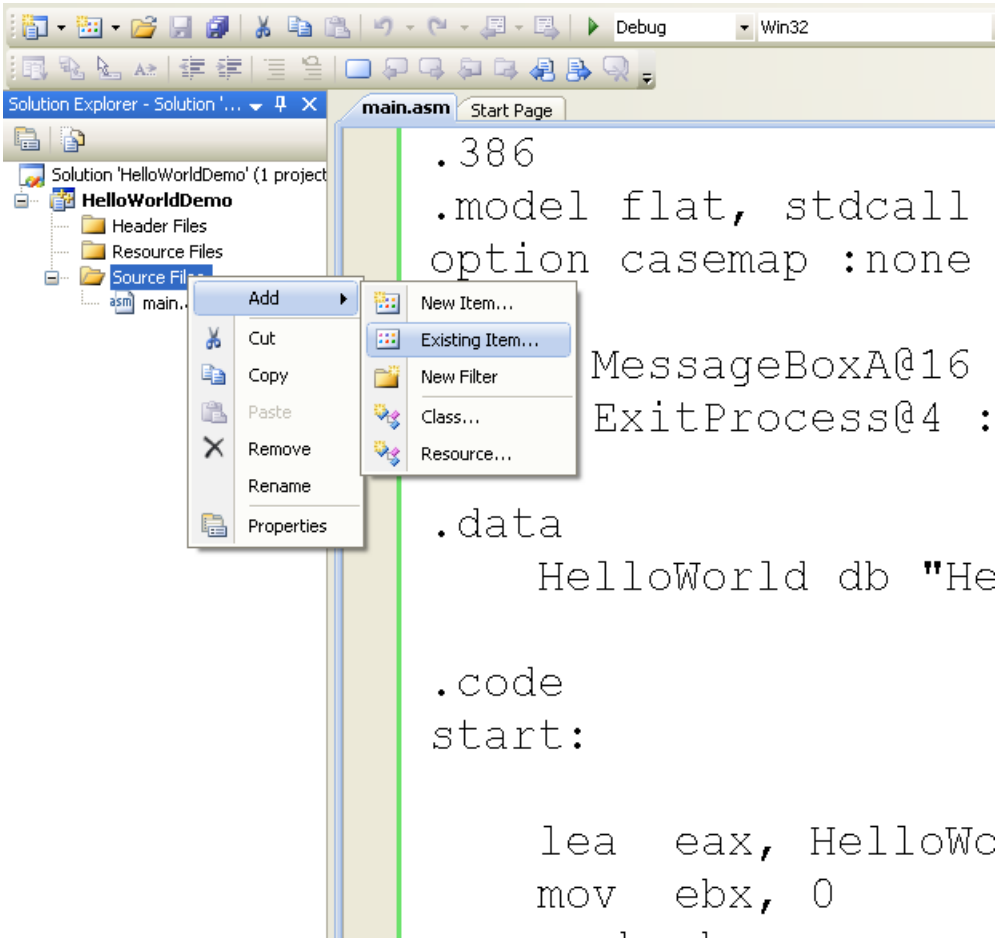
Then in the box titled Assembled Code Listing File, enter a file name. I chose asmListing File.

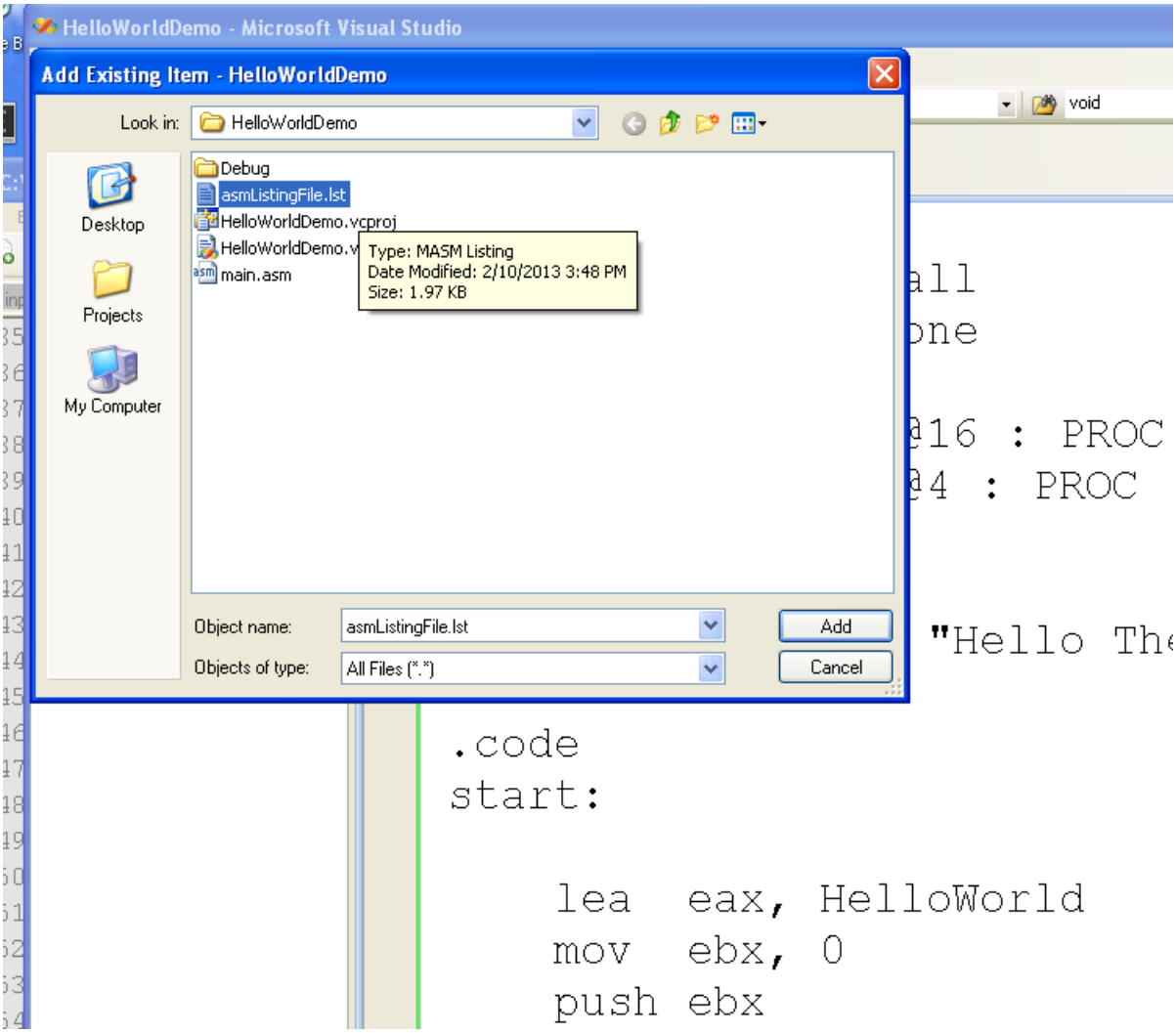


You will now have to Build the project again.



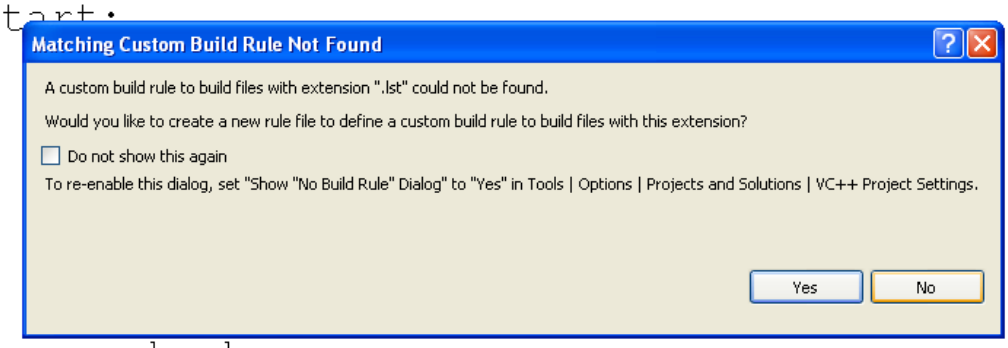
Right-click the folder called Source Files→Add→Existing Item…




HelloWorldDemo - Microsoft Visual Studio
Add Existing Item - HelloWorldDemo
Look in:
HelloWorldDemo
Desktop
Projects
My Computer
Debug
asmListingFile.lst
HelloWorldDemo.vcproj
main.asm
Type: MASM Listing
Date Modified: 2/10/2013 3:48 PM
Size: 1.97 KB
Object name:
asmListingFile.lst
Objects of type:
All Files (*.*)
Add
Cancel
void
all
one
@16 : PROC
@4 : PROC
"Hello The
.code
start:
lea eax, HelloWorld
mov ebx, 0
push ebx

Click No here:

```
extrn ExitProcess@4 : PROC

.data
    HelloWorld db "Hello There!",0

.code
start:
    push ebx
    call MessageBoxA@16
    push ebx
    call ExitProcess@4
```



The listing file allows you to see the exact hex values that your instructions get translated into:

```
00000000                .code
00000000                start:

00000000  8D 05 00000000 R       lea  eax, HelloWorld
00000006  BB 00000000            mov  ebx, 0
0000000B  53                  push ebx
0000000C  50                  push eax
0000000D  50                  push eax
0000000E  53                  push ebx
0000000F  E8 00000000 E          call MessageBoxA@16
00000014  53                  push ebx
00000015  E8 00000000 E          call ExitProcess@4
```

Here we see the instruction push ebx is represented by hex value 53.

In the next tutorial, I will show you a streamlined version of this in Cygwin.