

Using the Visual Studio Debugger

Introduction

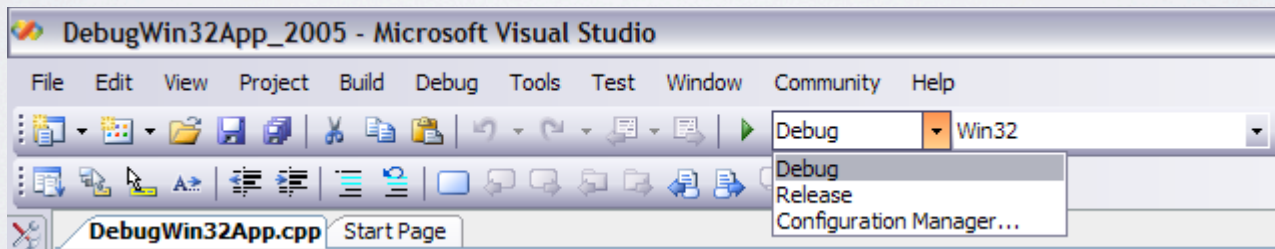
- There are two types of errors
 - Compilation errors
 - Logic errors (also called bugs)
- Eliminate compilation errors from your code
- Most C++ compiler vendors provide software called a debugger
 - Allows you to monitor the execution of your programs to locate and remove logic errors

Why Should I Use Visual Studio to Debug my Program?

- Even most experienced coder creates errors or “bugs”
- Visual Studio debugger will provide two powerful runtime facilities:
 - Trace the program Execution
 - Watch variables during program execution
- These allow you to stop at procedure locations, inspect memory and register values, change variables, observe message traffic, and get a close look at what your code does.

Project Configuration Settings

- Debug vs. Release Configurations
 - The **Debug** configuration of your program is compiled with full symbolic debug information and no optimization.
 - The **Release** configuration of your program is fully optimized and contains no symbolic debug information.
 - Must be in Debug configuration to debug your program!.



Getting Acquainted with Visual Studio Debugger

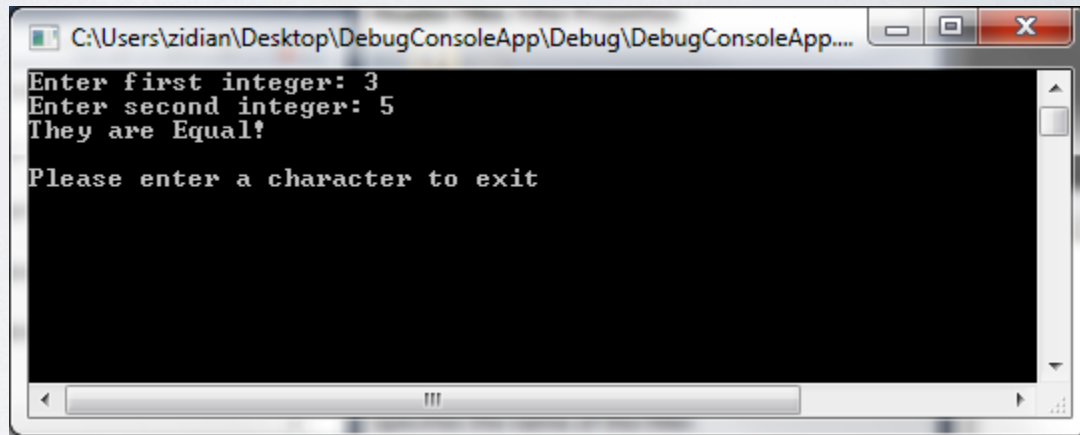
- Debugger Windows
 - Autos
 - Locals
 - Watch
 - Call Stack
 - etc.
- Execution Control
 - Starting or Continuing Execution
 - Stopping
 - Breaking Execution
 - Stepping Into and Out of code
 - etc.

Debugging Example #1

Console app

This simple console program should determine whether two integers are equal.

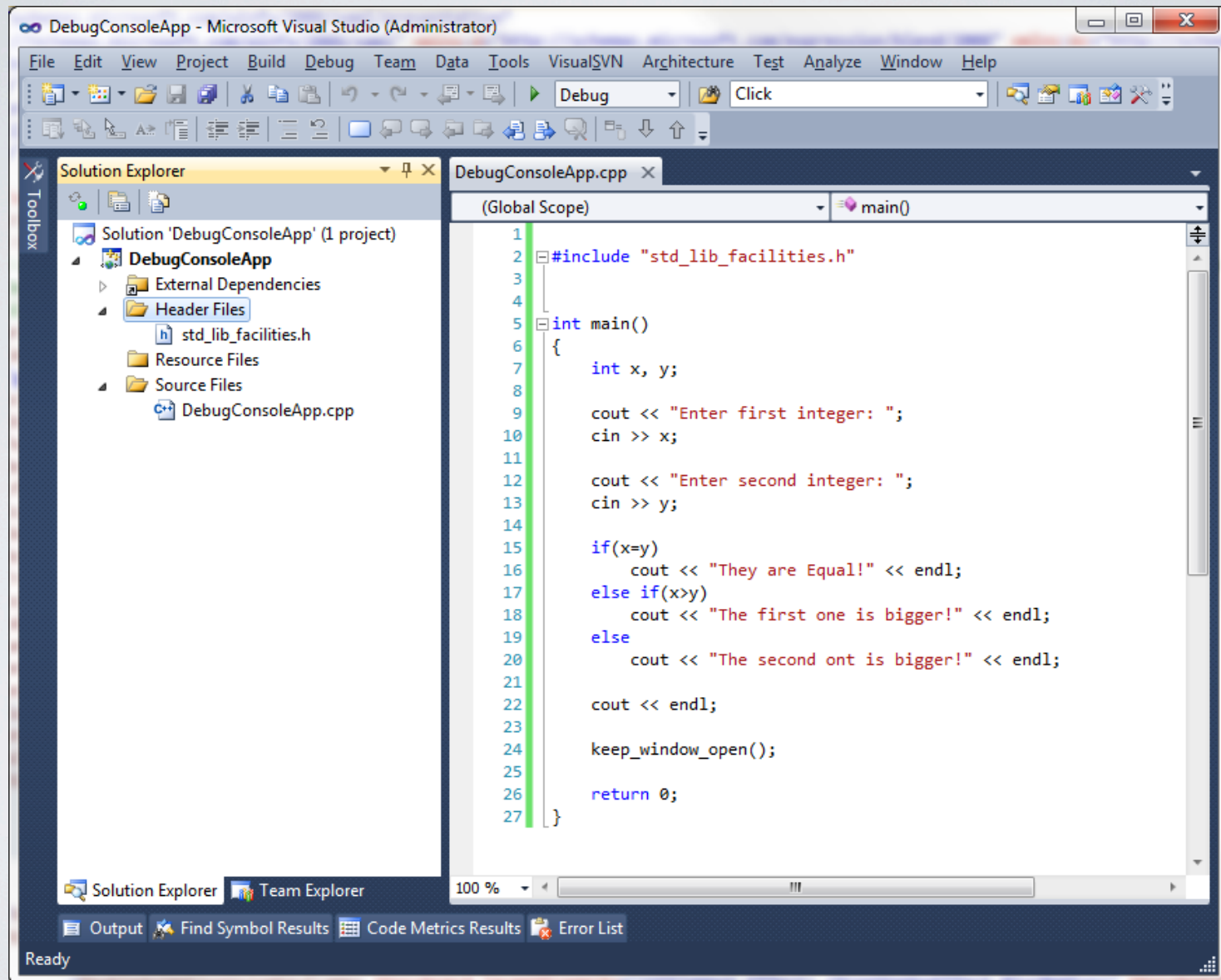
Code compiled just fine,
0 warnings, 0 errors



```
C:\Users\zidian\Desktop\DebugConsoleApp\Debug\DebugConsoleApp...  
Enter first integer: 3  
Enter second integer: 5  
They are Equal!  
  
Please enter a character to exit
```

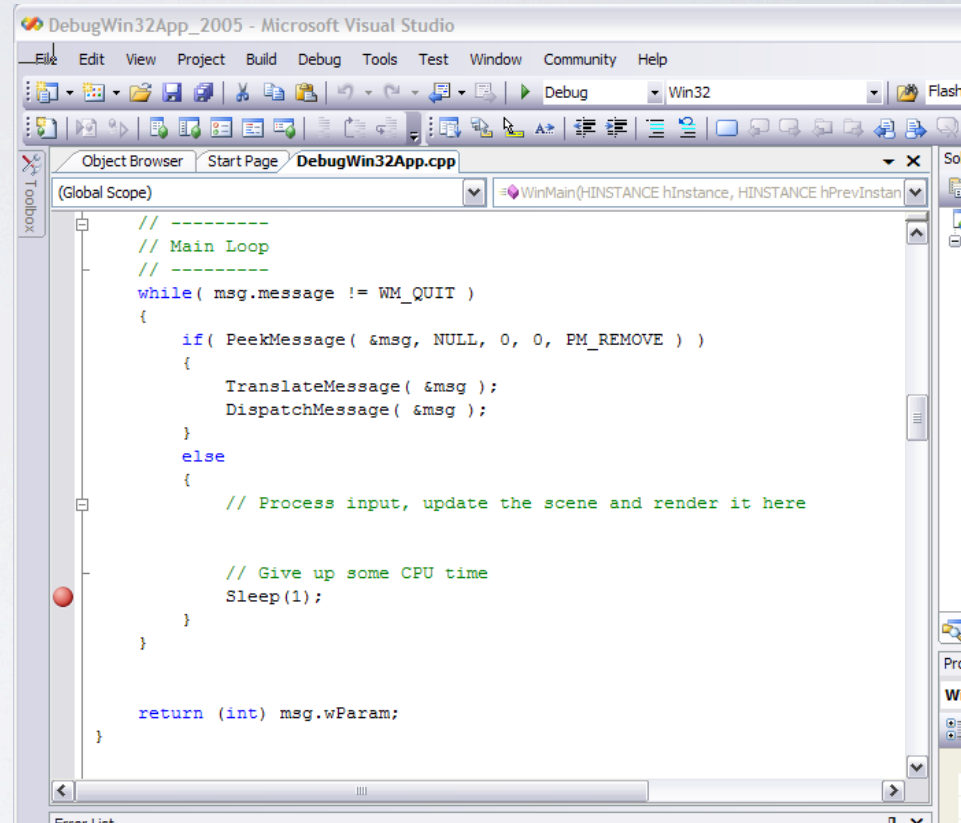
... BUT the code obviously has a logical error! 3 does not equal 5!

Debugging Example #1 (a console app.)

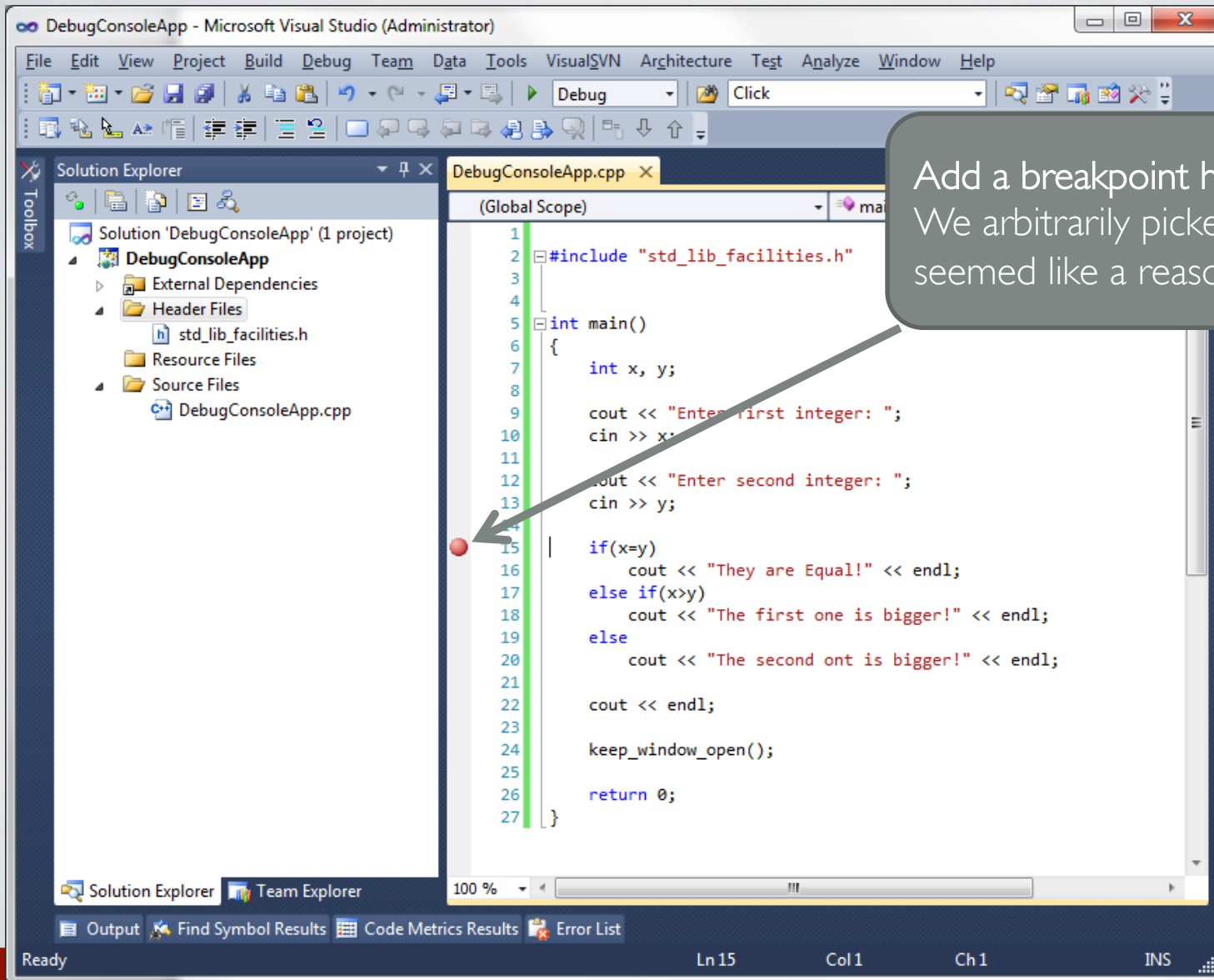


What is a Breakpoint?

- Breakpoints are user-defined code locations that pause execution
- You know them by the little, red “dot” in the left margin of the editor window
- F9 to add or remove (toggle)
- Or left-mouse click in margin
- Unlimited number of them to use.



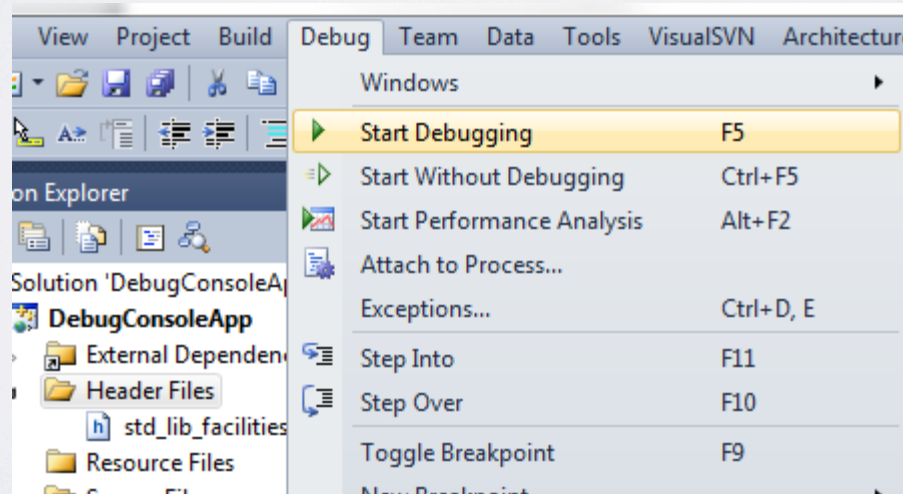
Debugging Example #1 (continued)



Add a breakpoint here
We arbitrarily picked this line because it
seemed like a reasonable place

Starting the Debugging Session

- Make sure you are in a Debug configuration
- Press F5
- Or click on Debug icon
- Or select menu Debug – Start Debugging



Debugging Example #1 - Running in the debugger

The screenshot shows the Microsoft Visual Studio IDE in Administrator mode, debugging a C++ application named 'DebugConsoleApp'. The main window displays the source code for 'DebugConsoleApp.cpp' at the 'main()' function. The code is as follows:

```
9 cout << "Enter first integer: ";
10 cin >> x;
11
12 cout << "Enter second integer: ";
13 cin >> y;
14
15 if(x=y)
16     cout << "They are Equal!" << endl;
17 else if(x>y)
18     cout << "The first one is bigger!" << endl;
19 else
20     cout << "The second one is bigger!" << endl;
21
22 cout << endl;
23
```

The debugger is paused at line 15. The 'Autos' window shows the current state of variables:

| Name | Value | Type |
|------|-------|------|
| x | 3 | int |
| y | 5 | int |

The 'Call Stack' window shows the following stack frames:

| Name | Language |
|------------------------------------------------------------------------------------|----------|
| DebugConsoleApp.exe!main() Line 15 | C++ |
| DebugConsoleApp.exe!_tmainCRTStartup() Line 555 + 0x19 bytes | C |
| DebugConsoleApp.exe!mainCRTStartup() Line 371 | C |
| kernel32.dll!7526339a() | |
| [Frames below may be incorrect and/or missing, no symbols loaded for kernel32.dll] | |
| ntdll.dll!776d9ef2() | |
| ntdll.dll!776d9ec5() | |

The bottom status bar shows 'Ready'.

Debugging Example #1 Stepping, examine variables

The screenshot shows the Microsoft Visual Studio IDE in Administrator mode, debugging a console application named 'DebugConsoleApp'. The code in the main() function is as follows:

```
9 cout << "Enter first integer: ";
10 cin >> x;
11
12 cout << "Enter second integer: ";
13 cin >> y;
14
15 if(x=y)
16     cout << "They are Equal!" << endl;
17 else if(x>y)
18     cout << "The first one is bigger!" << endl;
19 else
20     cout << "The second ont is bigger!" << endl;
21
22 cout << endl;
23
```

The debugger is currently stopped at line 16. The Autos window shows the following variables:

| Name | Value | Type |
|------|-------|------|
| x | 5 | int |
| y | 5 | int |

The Call Stack window shows the following stack frames:

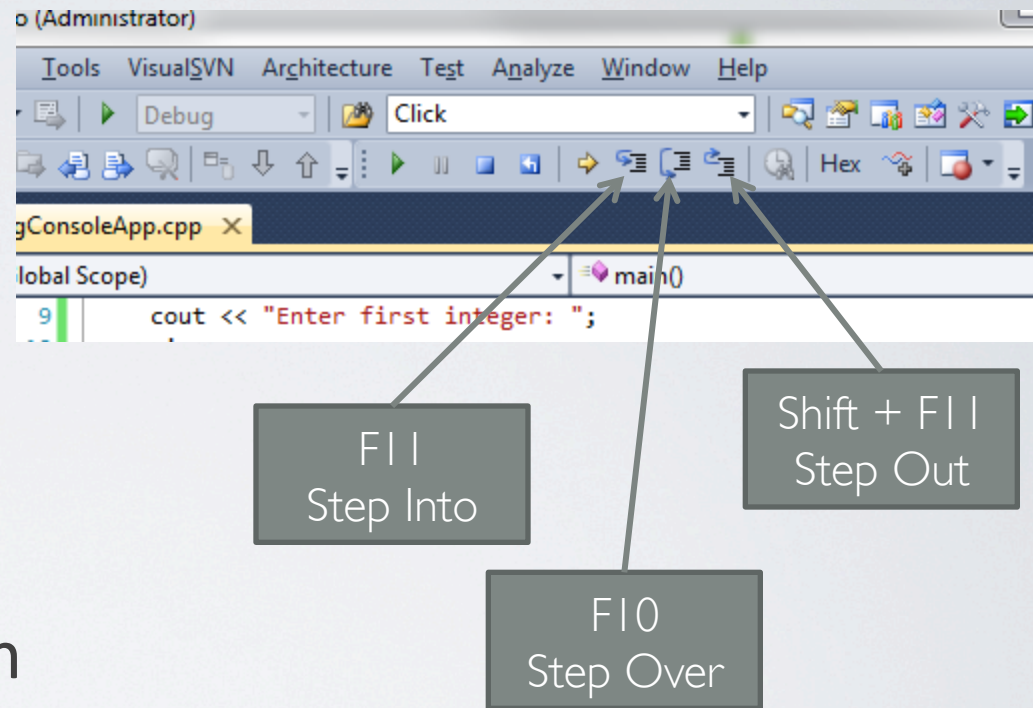
| Name | Language |
|-----------------------------------------------------------------------------------------------|----------|
| DebugConsoleApp.exe!main() Line 16 | C++ |
| DebugConsoleApp.exe!_tmainCRTStartup() Line 555 + 0x19 bytes | C |
| DebugConsoleApp.exe!mainCRTStartup() Line 371 | C |
| kernel32.dll!7526339a() | |
| [Frames below may be incorrect and/or missing, no symbols loaded for kernel32.dll!776d9ef2()] | |
| ntdll.dll!776d9ec5() | |

The status bar at the bottom indicates the system is 'Ready' and the current cursor position is 'Ln 16 Col 1 Ch 1'.

Execution Control

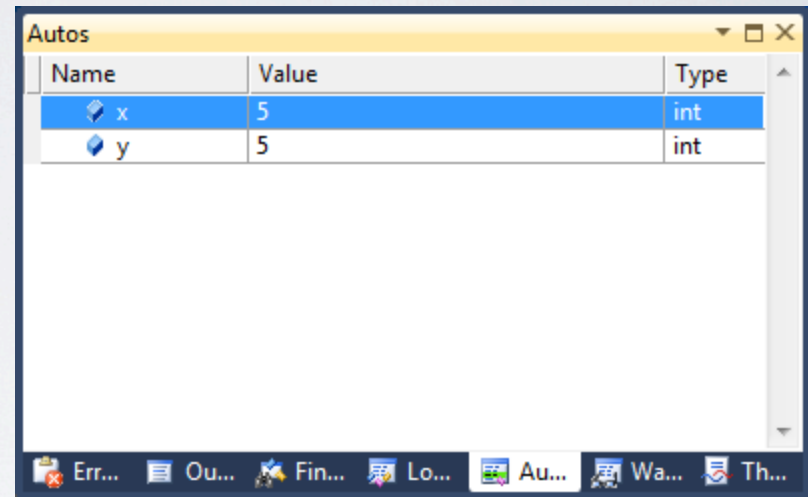
Stepping through your code

- Starting / Stopping
- Breaking
- Stepping through your application
 - – (F10, F11 or Toolbar buttons)
- Run to a specific location
- **Run To Cursor** (right-click menu)



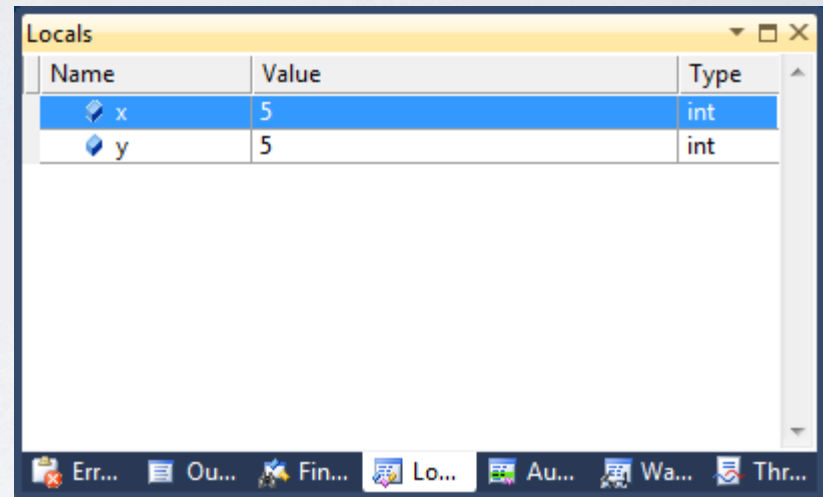
Autos Window

- **Name**
 - The names of all variables in the current statement and the previous statement. The current statement is the statement at the current execution location, which is the statement that will be executed next if execution continues.
- **Value**
 - The value contained by each variable. By default, integer variables are represented in decimal form.
- **Type**
 - The data type of each variable listed in the Name column.



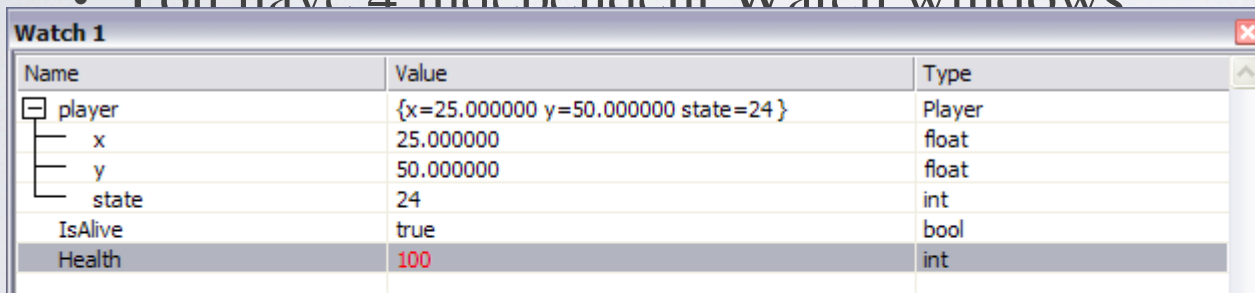
Locals Window

- **Name**
 - This column contains the names of all local variables in the current scope.
- **Value**
 - The value contained by each variable. By default, integer variables are represented in decimal form.
- **Type**
 - The data type of each variable listed in the **Name** column.



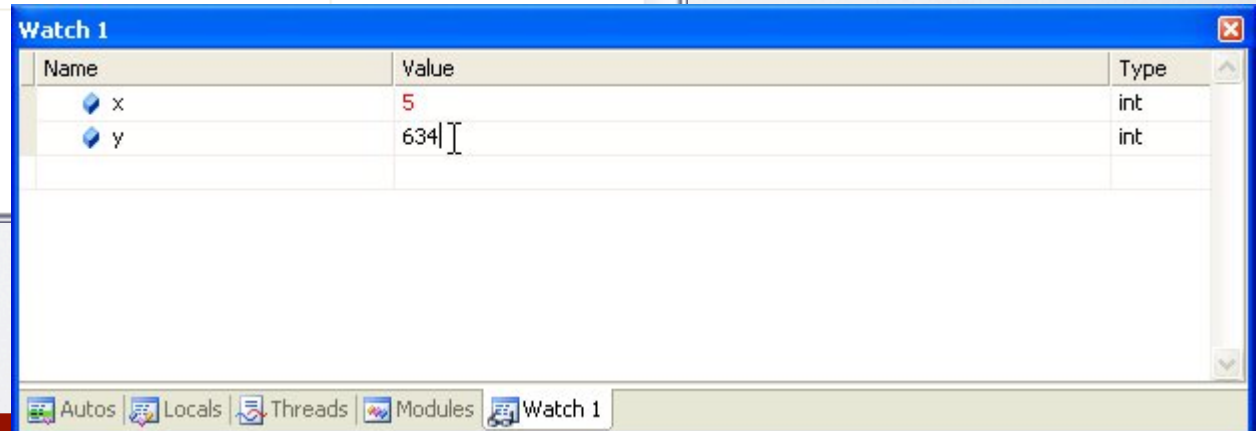
Watch window(s)

- Watch window displays Name, Value, and Type of variables
- Type in or click-drag variables into window
- Change values live while at break
- You have 4 independent Watch windows



Watch 1

| Name | Value | Type |
|---------|-------------------------------------|--------|
| player | {x=25.000000 y=50.000000 state=24 } | Player |
| x | 25.000000 | float |
| y | 50.000000 | float |
| state | 24 | int |
| IsAlive | true | bool |
| Health | 100 | int |



Watch 1

| Name | Value | Type |
|------|-------|------|
| x | 5 | int |
| y | 634 | int |

Autos Locals Threads Modules Watch 1

Debugging Example #1 – Found error

The screenshot shows the Microsoft Visual Studio IDE in Administrator mode, debugging a C++ application named 'DebugConsoleApp'. The code in 'DebugConsoleApp.cpp' is as follows:

```
9 cout << "Enter first integer: ";
10 cin >> x;
11
12 cout << "Enter second integer: ";
13 cin >> y;
14
15 if(x=y)
16     cout << "They are Equal!" << endl;
17 else if(x>y)
18     cout << "The first one is bigger!" << endl;
19 else
20     cout << "The second one is bigger!" << endl;
21
22 cout << endl;
23
```

A callout box highlights the error: `if (x==y)`. The debugger's Autos window shows the current state of variables:

| Name | Value | Type |
|------|-------|------|
| x | 3 | int |
| y | 5 | int |

The Call Stack window shows the current execution point at line 15 of 'DebugConsoleApp.exe!main()'. The stack includes:

- DebugConsoleApp.exe!main() Line 15 (C++)
- DebugConsoleApp.exe!_tmainCRTStartup() Line 555 + 0x19 bytes (C)
- DebugConsoleApp.exe!mainCRTStartup() Line 371 (C)
- kernel32.dll!7526339a()
- [Frames below may be incorrect and/or missing, no symbols loaded for kernel32.dll]
- ntdll.dll!776d9ef2()
- ntdll.dll!776d9ec5()

Debugging Example #1 - Fixed error, recompiled, run, step

The screenshot shows the Microsoft Visual Studio IDE in Administrator mode, debugging a C++ application named DebugConsoleApp. The main window displays the source code for DebugConsoleApp.cpp, with the debugger paused at line 17. The code is as follows:

```
10 cin >> x;
11
12 cout << "Enter second integer: ";
13 cin >> y;
14
15 if(x==y)
16     cout << "They are Equal!" << endl;
17 else if(x>y)
18     cout << "The first one is bigger!" << endl;
19 else
20     cout << "The second ont is bigger!" << endl;
21
22 cout << endl;
23
24 keep_window_open();
```

The Autos window shows the current state of variables:

| Name | Value | Type |
|------|-------|------|
| x | 3 | int |
| y | 5 | int |

The Call Stack window shows the current function and its callers:

| Name | Language |
|------------------------------------------------------------------------------------|----------|
| DebugConsoleApp.exe!main() Line 17 | C++ |
| DebugConsoleApp.exe!_tmainCRTStartup() Line 555 + 0x19 bytes | C |
| DebugConsoleApp.exe!mainCRTStartup() Line 371 | C |
| kernel32.dll!7526339a() | |
| [Frames below may be incorrect and/or missing, no symbols loaded for kernel32.dll] | |
| ntdll.dll!776d9ef2() | |
| ntdll.dll!776d9ec5() | |

The bottom of the window shows the Windows taskbar with several open applications and the system tray displaying 'Ready'.

Debugging Example #1 - Step. Hey the code worked!

The screenshot shows the Microsoft Visual Studio (Administrator) interface. The main window displays the source code for `DebugConsoleApp.cpp` in the `main()` function. The code is as follows:

```
10 cin >> x;
11
12 cout << "Enter second integer: ";
13 cin >> y;
14
15 if(x==y)
16     cout << "They are Equal!" << endl;
17 else if(x>y)
18     cout << "The first one is bigger!" << endl;
19 else
20     cout << "The second ont is bigger!" << endl;
21
22 cout << endl;
23
24 keep_window_open();
```

The debugger has stopped at line 20, indicated by a yellow arrow and a red dot in the left margin. The `Call Stack` window at the bottom right shows the following frames:

| Name | Language |
|------------------------------------------------------------------------------------|----------|
| DebugConsoleApp.exe!main() Line 20 | C++ |
| DebugConsoleApp.exe!_tmainCRTStartup() Line 555 + 0x19 bytes | C |
| DebugConsoleApp.exe!mainCRTStartup() Line 371 | C |
| kernel32.dll!7526339a() | |
| [Frames below may be incorrect and/or missing, no symbols loaded for kernel32.dll] | |
| ntdll.dll!776d9ef2() | |
| ntdll.dll!776d9ec5() | |

The `Autos` window at the bottom left is empty. The `Immediate Window` is also empty. The status bar at the bottom shows "Ready", "Ln 20", "Col 1", "Ch 1", and "INS".