Module 02 Lab 02 Worksheet

Debugging C# in Visual Studio Code

# Overview

Debugging is a useful skill for any programmers. In addition to being able to "hand-debug" your code, you should also get to know the debugger tools for your platform and language.

Out of the box, VS Code has debugging support for Javascript. Support for debugging additional languages is acquired with the use of extensions. The C# extension that we installed when we set up our developer workspace not only includes Intellisense and syntax highlighting but also debugging for C# applications.

|  |
| --- |
| **NOTE**: This lab assumes you've done Lab 01 in this module and have created a C# project called VarmintHunt. In addition, you may wish to read Lecture 02 on debugging to get background on the VS Code debugger interface. |

|  |  |
| --- | --- |
| * Start VS Code. * Open the VarmintHunt project you created in the previous lab. | 0 |
| In Program.cs, set a breakpoint at the start of the try block.  Click on the Debug icon in the Side Bar.  Press F5 or click on Start Debugging (the green triangle at the top of the Debugging View Bar.  How many variables are listed in the Locals section of the Variables window? (5 pts.)  four.  List their names , data types and current values. ( 5 pts.)  args string[] {string[0]}  fred, Varmint null  sam VarmintHunter null  trouble int[] null | 10 |
| Press F5 or click on Continue in the Debug panel.  What variables are show in the Variables window? (5 pts.)  What happened? (5 pts.) | 10 |
| Insert another breakpoint just before the first Console.Writeline() in Main().  Start the debugger.  When it stops at the first breakpoint, click on Continue.  What happens? (5 pts.)  Stop the debugger.  Go to your first breakpoint, right-click on it and select Disable Breakpoint.  Restart the debugger.  Where does program execution stop? (5 pts.)  You can have multiple breakpoints in your code and then use enable/disable to help you move around faster. | 10 |
| Pick one of your class files and set a breakpoint inside of one of your class methods.  Make sure that your Main() method invokes that method at least once.  Restart the debugger.  Where does execution stop? (5 pts.)  Click on Continue.  Where does execution stop? (5 pts.)  You can set breakpoints in other code files in your project. This makes it easier to follow execution as it goes from one context to another. | 10 |
| Remove the breakpoint from your class file.  Start the debugger again.  Press F10 or click on Step Over in the Debug panel.  At what line number does code execution stop? (5 pts.)  Press F10 or click on Step Over once more.  One of the variables in the Variables window has changed. Which one? (5 pts.) | 10 |
| Press F10 or click on Step Over once more.  What happens to the variable that changed in the previous step? (5 pts.)  Stop the debugger.  Set a breakpoint just before the Varmint object declaration.  Keep pressing F10 until you come to the line just after the declaration of the Varmint object.  In the Variables window, how has the Varmint variable changed? (5 pts.) | 10 |
| Click on Step Over again or press F10.  What has changed in the Variables window? (5 pts.)  The VarmintHunter object now has a Photos property with the value of 0.  Press F10 again. How has the variable changed? (5 pts.) | 10 |
| Keep pressing F10 until your program completes.  In the Breakpoints window on the left, right-click and select Remove All Breakpoints.  Add the following lines to the beginning of your Main method (just before the try block):  for(int i = 0; i < 10; i++)  Console.WriteLine("i is " + i);  Set a new breakpoint at the start of the for loop.  Restart the debugger.  Press F10 six times. What is the output at the Debug console? (the dark colored text) (5 pts.)  Press F11 or click on Step Out Of. What happens? (you may have to scroll the Debug Console window up a bit) (5 pts.)  "Step Over" will step through the program line-by-line but not follow any function calls that are out of the current context.  "Step In" will follow the code line-by-line even if execution moves outside of the main function.  "Step Out" returns you to wherever you "stepped in".  By using these three controls along with appropriate breakpoints, you can step in and out of any section of your program that you wish.  So far it still means that you have to go step-by-step through a loop until the variable inside takes on the value you're interested in. *But what if it didn't?* | 10 |
| Stop the debugger by clicking on the red stop button on the Debug bar. (Or by pressing <Shift>-F5)  Remove the previous breakpoint and place one just inside the for loop.  Run the debugger. Note how the breakpoint triggers every time the loop runs.  What if we wanted to have it stop when the variable i reached a certain value?  Stop the debugger.  Right-click on the breakpoint and select Edit Breakpoint.  In the box that appears, enter  i == 4  And press <Enter>  How has the breakpoint changed? (5 pts.)  Restart the debugger.  What is the value of i in the Variables window? (5 pts.)  Now if you use "Step Over" or "Step In", the loop will proceed one iteration at a time from that point. | 10 |
| Stop the debugger.  The Watch window will let us check the state of variables whether they are in or out of scope. (Variables only looks at the current scope).  In the Watch window title bar, click on the plus sign (+), type the letter i and press <Enter>.  What is the value shown in the Watch window for this variable? (5 pts.)  Start up the debugger again.  When it stops at our conditional breakpoint, what is the value of i in the Watch window? (5 pts.)  In the Variables window? (5 pts.) | 10 |
| The lab is now complete.   * Stop the debugger. * Remove all of the breakpoints. * Close your project. | 0 |
| **Total** |  |

# Summary

This lab introduced you to the basics of the built-in debugger in Visual Studio code. You learned:

* How to set and edit breakpoints
* How to monitor variables in the Variables window
* How to use the Watch window
* How to navigate through your code with Step In, Step Over, Step Out and Continue.

Complete this worksheet and submit it to your instructor.