VS Debugger

Contents

[1 Breakpoints 2](#_Toc127020551)

[2 Examining the Value of Variables 3](#_Toc127020552)

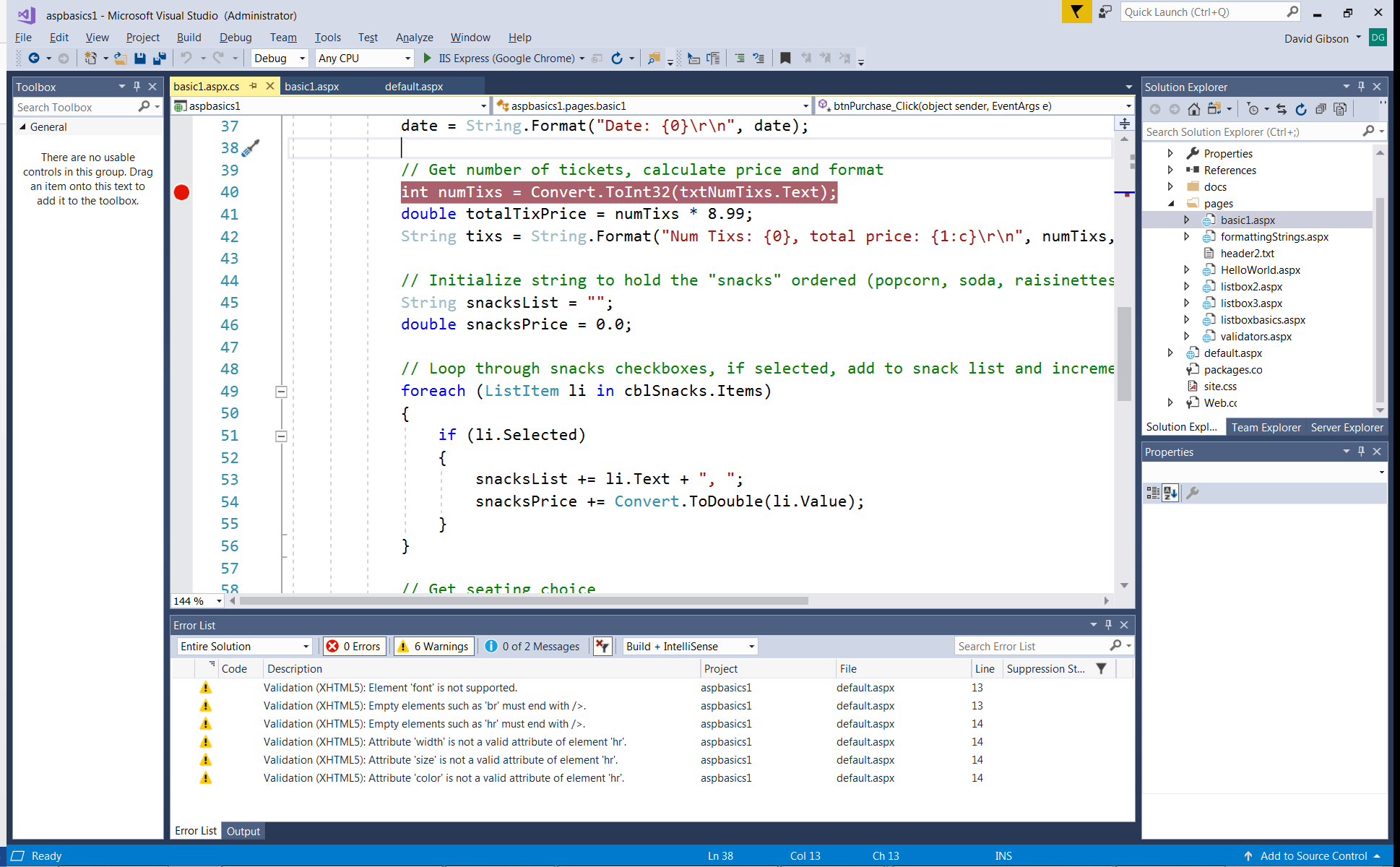
[3 Step through code 4](#_Toc127020553)

[4 Occasional Use 4](#_Toc127020554)

[Appendix 1 n/a 4](#_Toc127020555)

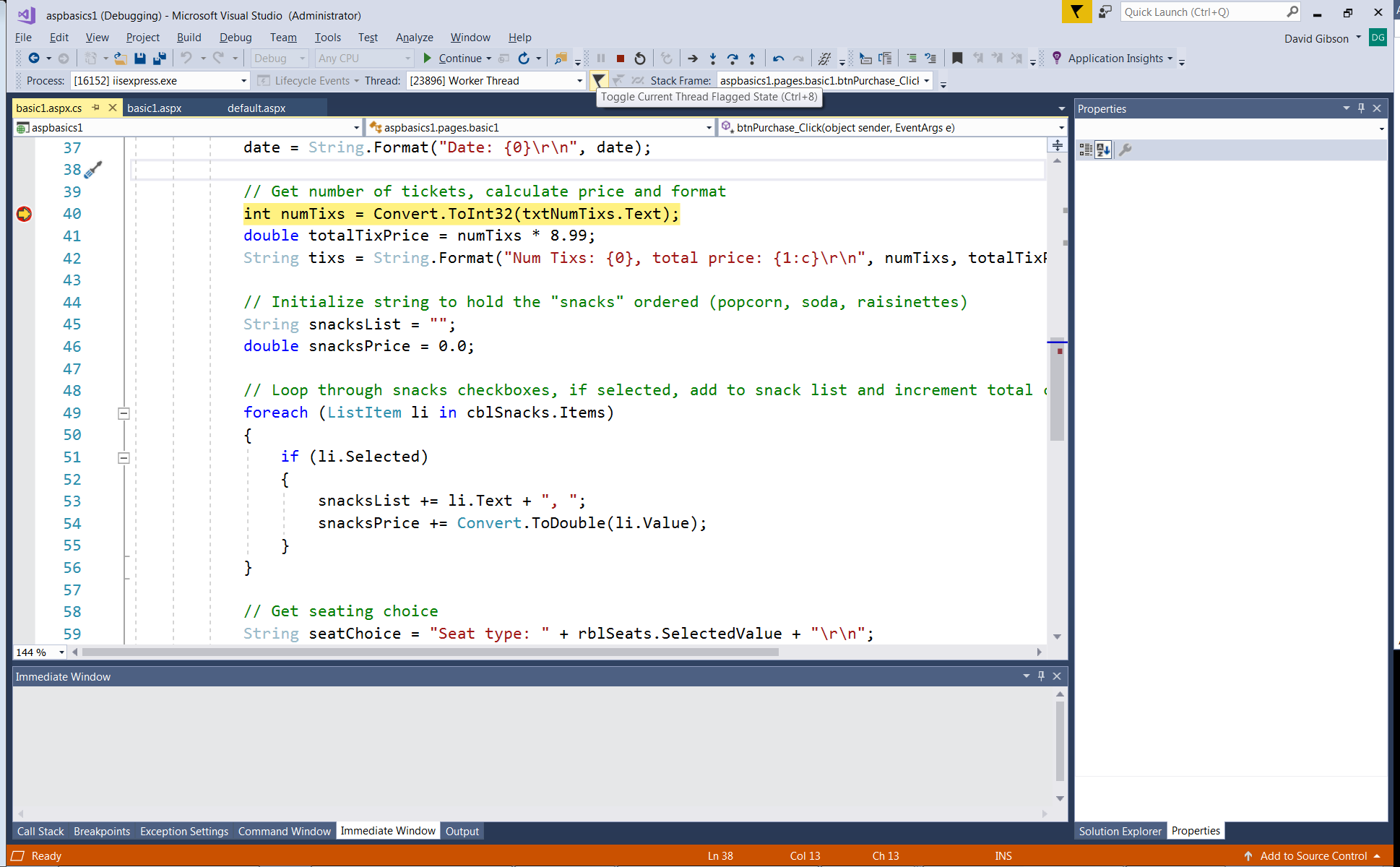
# Breakpoints

These steps illustrate how to use the debugger.

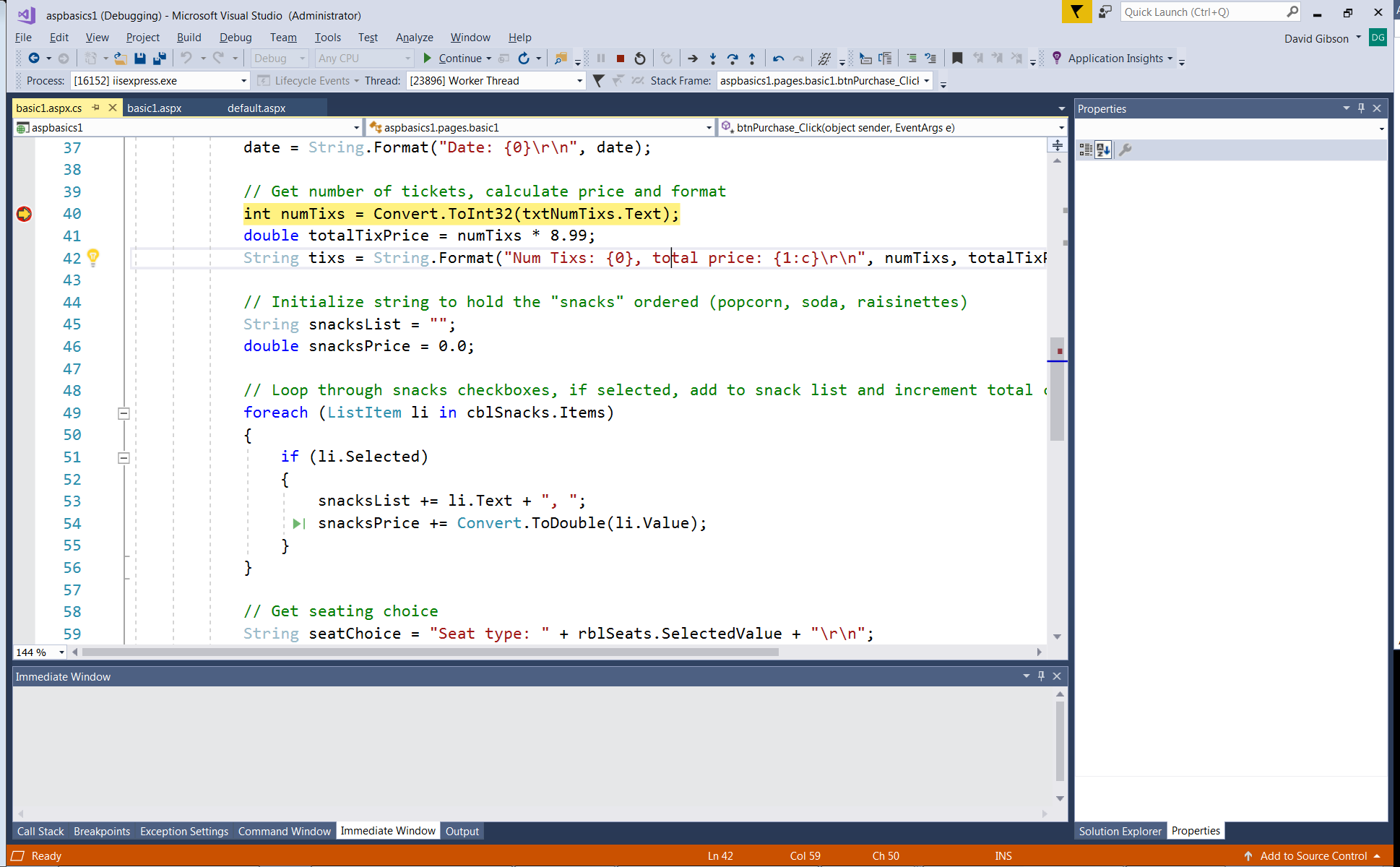
1. **Set a breakpoint** – A breakpoint is a line of code where we want to stop execution of the code and examine the value of a variable(s). To set a breakpoint,
2. Find the line of code where you want to stop execution
3. Click in the vertical gray bar on the far left of the code window.

Notes:

* You can set as many breakpoints as you want which will stop execution at each of them.
* You can remove a breakpoint by double-clicking the red circle.
* You can remove all breakpoints by choosing: Debug, Delete All Breakpoints (Ctrl+Shift+F9).

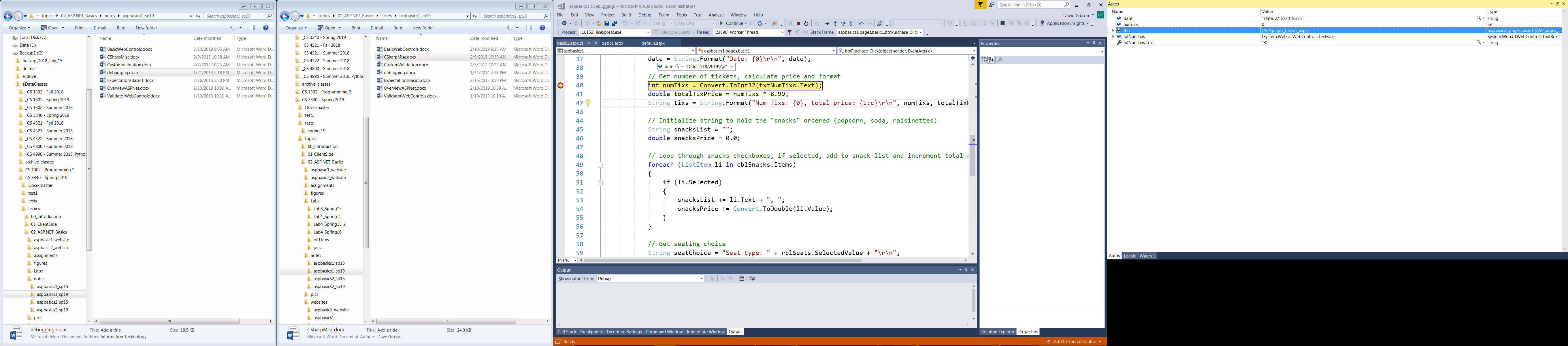
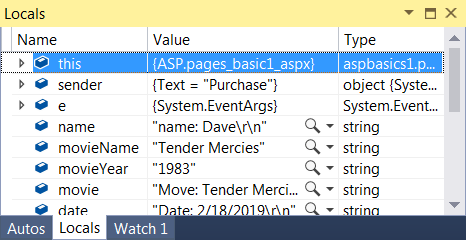
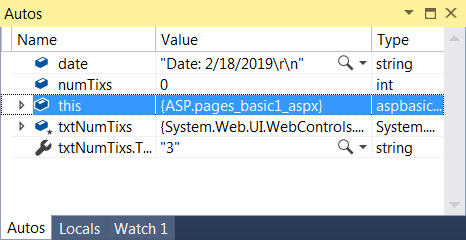
1. **Run the app until the breakpoint is reached** – Choose: Debug/Start Debugging (or F5). The code will run to this point and stop. The line is now highlighted yellow.

Notes:

* The yellow highlighted line has not been executed. It is the next line that will be executed.
* Probably your breakpoint is an event handler that is not called the first time the page is displayed. Thus, the web page will be displayed. Then, you perform some action that causes a postback and then the code will stop and return back to VS. When you start the debugger, it will usually display the browser. Sometimes you may need to switch over to it. Similarly, when you cause a postback, it should switch back to VS. Sometimes you may need to switch yourself back.
* Sometimes you may wonder, for any number of reasons, if the debugger is running. It is running if you see a yellow highlighted line of code. However, if you don’t, here are two (identical) ways to determine that the debugger is running:

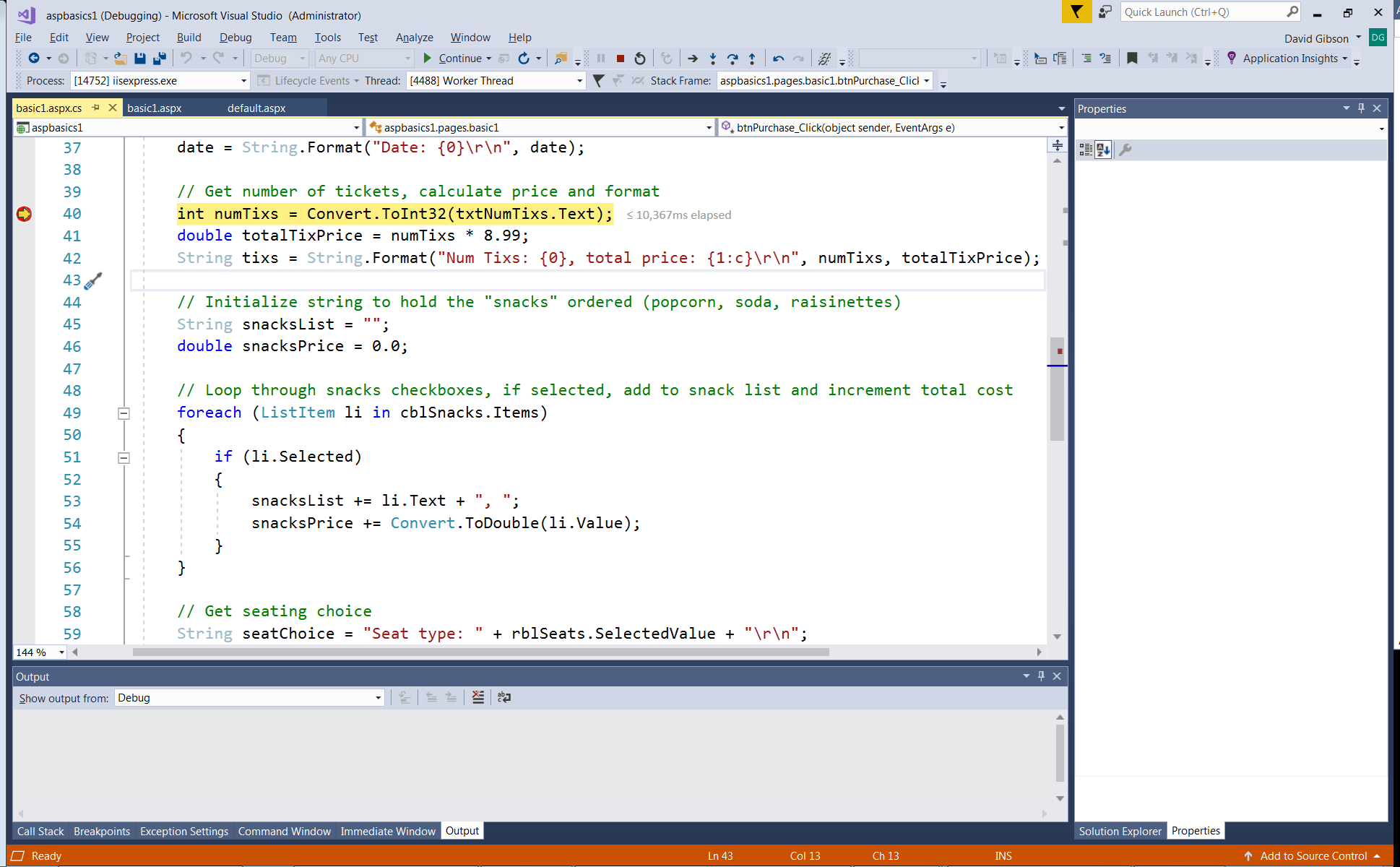
1. There is a maroon square on the Ribbon as shown on the right
2. Check the Debug menu, if you see *Stop Debugging*

# Examining the Value of Variables

1. **Examine the value of variables** – There are three ways:
2. Hover your mouse over a variable. A floating display will show the value.
3. Display the Locals window. Choose: Debug, Windows, Locals if necessary. The Locals window shows variables defined in the local scope, which is usually the current function or method, and their current value.
4. Display the Autos window. Choose: Debug, Windows, Autos if necessary. The Autos window shows variables used around the current breakpoint. Notes:

* *this* is the page object itself. When expanded, it shows every property (including controls) defined in the page. The first 100+ properties are inherited from the *Page* superclass.
* The Locals window is usually what you want to use when debugging. However, sometimes the higher level view of the Autos window is useful.

# Step through code

1. **Step through code** – Choose one of the options below. Note: All of the options below are available on the Debug panel on the Ribbon. If you don’t see this, then right-click the Ribbon and choose: Debug.
2. Step over the current line of code – Choose: Debug, Step Over (or F10).

Executes the highlighted line of code. If it is a method call, then it executes the method without entering it.

1. Step into a method call – Choose: Debug, Step Into (or F11).

If the highlighted line is a method call then you will enter the method.

1. Step out of a method call – Choose: Debug, Step Out (or Shift+F11)

Returns to the calling method.

1. Continue running – Choose: Debug, Continue (or F5)

Runs to the next breakpoint, displaying the browser if needed. If there are no more breakpoints, it runs until the first breakpoint is reached.

1. Step backwards – Choose: Debug, Step Backwards

Once you have stepped forward beyond the breakpoint, you can step backwards until you reach the breakpoint.

1. Stop debugging – Choose: Debug, Stop Debugging (or Shift+F5)

# Occasional Use

1. Immediate Window – You can type a code statement directly into this window and execute it by pressing Enter.

Example Use Case: Suppose you have stopped execution at some place in your code. You examine the value of a variable, *x* and observed that it is incorrect. You suspect that you didn’t divide it by a variable, *y* earlier. You can simply type: “x/y” into the Immediate Window and press Enter to see the new value.

If the Immediate Window is not visible, choose: Debug, Windows, Immediate.

1. Set a Watch Variable(s). You can set variables to watch in the Watch Window (Debug, Windows, Watch). I believe the way this works is that while in the Debugger, the code will stop every time a watched variable changes value. See [Watch Variables](https://docs.microsoft.com/en-us/visualstudio/debugger/watch-and-quickwatch-windows?view=vs-2017) for more information.

Appendix

1. n/a