

CSCI 1301 – Week 1 - Lab

August 13, 2020

1 Welcome!

For this first (virtual!) lab you will be given instructions to perform simple steps to set up a working environment and the course administrators' contact information to make sure you know how to reach your instructors. This first part will address three of your instructor and undergraduate teaching assistant needs:

1. We would like to get to know you.
2. We need to know which lab session is the most suited for you.
3. We need to make sure you know how to reach out to us.

For the first two parts, please complete the survey on D2L called “Getting to know you + Pick your lab!”. For the last part, make sure you’ve read Dr. Aubert’s email “CSCI 1301 – Second contact” sent on 08/10, and that you have access to the “CSCI 1301 – Fall 2020 – Section A & B” team on Teams. Don’t hesitate to say, “hi” in the chat room!

2 Accessing And Using Visual Studio

The software we will primarily be using this Semester is called Visual Studio. This part will detail how to access it. You will need either

- a computer with the right to install software on it,
- to access one of the computers in the computer lab¹, or
- a computer with internet access.

The third solution is a backup plan, as instead you will access the minimal version of Visual Studio to test small snippets of code. You should not rely on it for the duration of this course.

2.1 Installing Visual Studio On Your Own Computer

This part gathers some references for you to install Visual Studio on your own computer, regardless of your operating system. Note that we are *not* installing “Visual Studio *code*”, but simply “Visual Studio”. It is strongly encouraged that you do so, especially if you want to continue in a CS/IT/Cyber degree, but is not mandatory². The instructions are not very detailed: feel free to look for set-up help on the internet, from your classmates, and from your instructors.

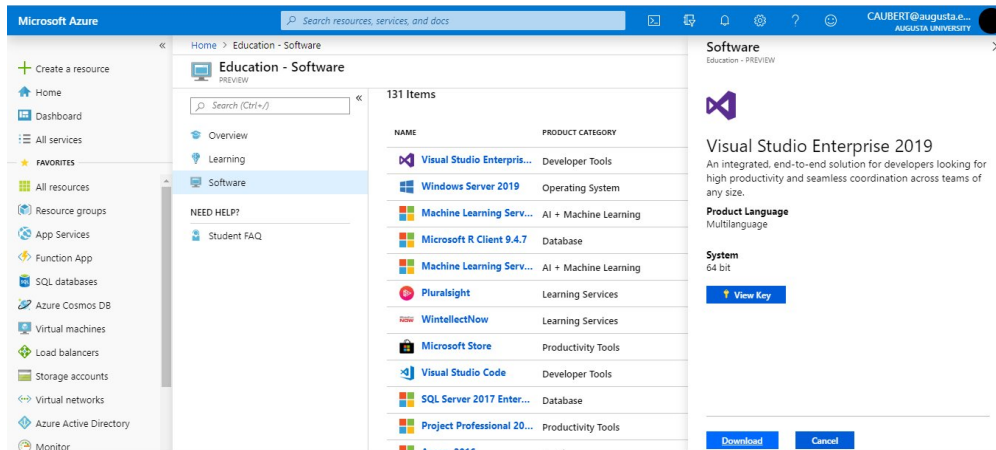
There are plenty of ways this can go wrong: make sure you have read and followed those instructions carefully before asking for help!

¹<https://www.augusta.edu/its/computers-printing.php>

²Unless we have to move fully online at some point in the semester.

2.1.1 For Windows

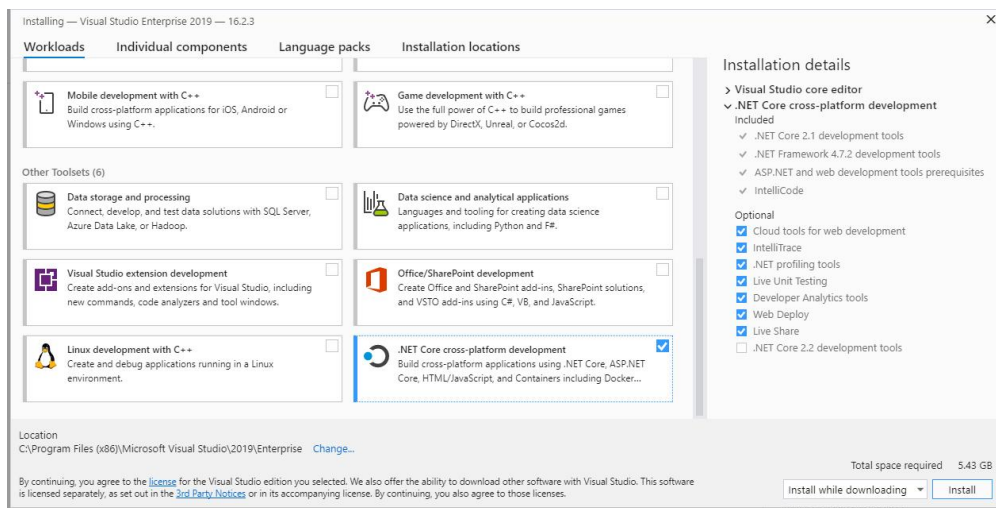
1. Visit Azure Dev Tools for Teaching³.
2. Log in using your Augusta University credentials.
3. Select “Download software”.
4. Look for Visual Studio. The path is Education → Software → Visual Studio Enterprise 2019 . You can search “Services” for the “Education” group and then click “Software” if the education group is not immediately displayed. It should look like the following:



Normally, the following direct link should get you to the right page: https://portal.azure.com/?Microsoft_Azure_Education_correlationId=8ee63052-dc32-46f7-a109-e26793622dbf#. Type “Education” in the search bar to get started on the path to the program to download.

5. Download and install Visual Studio (leave all the options on their default settings).

If possible, check the box next to “Workload” → “Windows” → “.NET-Desktop Development”, or to “Workload” → “Other Toolsets” → “.NET Core cross-platform development”:



³<https://aka.ms/devtoolsforteaching>

6. Enter the product key you obtained previously, following the instructions in the documentation⁴. Normally, clicking on “View key” on the screen pictured in the fourth step above should give you access to a key, that you simply need to copy-and-paste in the menu you can access on Visual Studio by clicking on “Select File” → “Account Settings” → “License with a Product Key”.

2.1.2 For Mac and Other Linux Systems

You can either install another version of Visual Studio or “emulate” Windows on your computer.

For the first option, download a version of Visual Studio at <https://visualstudio.microsoft.com/vs/mac/>. It differs a bit from the windows version, but that should not impact your experience in this class.

For the second option, you will need a Virtual Machine⁵ manager. This option works for Linux systems as well.

1. You can use
 - a) “VMware Fusion 10.x Pro” (only for MacOS, available on OnTheHub⁶,
 - b) Virtual Box⁷ (for Linux and Mac),
 - c) Virtual Machine Manager⁸ (for Linux).
2. Download a version of “Microsoft Operating Systems” from Azure Dev Tools for Teaching⁹,
3. Install and run your version of Windows from your virtual machine, and follow [the instructions for windows](#) to install Visual Studio.

2.2 Accessing One of the Computers in a Computer Lab

Please refer to this page from AU’s Information Technology¹⁰ to know where the computer labs are located. Visual Studio should be pre-installed on every computer.

2.3 Compiling Code On-Line

As a backup or only to test snippets of code, you can compile C# code online. Multiple online platforms exist, such as:

- <https://rextester.com/>
- https://www.tutorialspoint.com/compile_csharp_online.php
- https://www.onlinegdb.com/online_csharp_compiler
- <https://www.jdoodle.com/compile-c-sharp-online/>
- <https://dotnetfiddle.net/>

⁴<https://docs.microsoft.com/en-us/visualstudio/ide/how-to-unlock-visual-studio?view=vs-2019>

⁵https://en.wikipedia.org/wiki/Virtual_machine

⁶<https://e5.onthehub.com/WebStore/OfferingDetails.aspx?o=637dd37b-06b5-e711-80f7-000d3af41938&pmv=00000000-0000-0000-0000-000000000000&ws=2020165a-723a-de11-b696-0030485a8df0&vsro=8>

⁷<https://www.virtualbox.org/>

⁸<https://virt-manager.org/>

⁹<https://aka.ms/devtoolsforteaching>

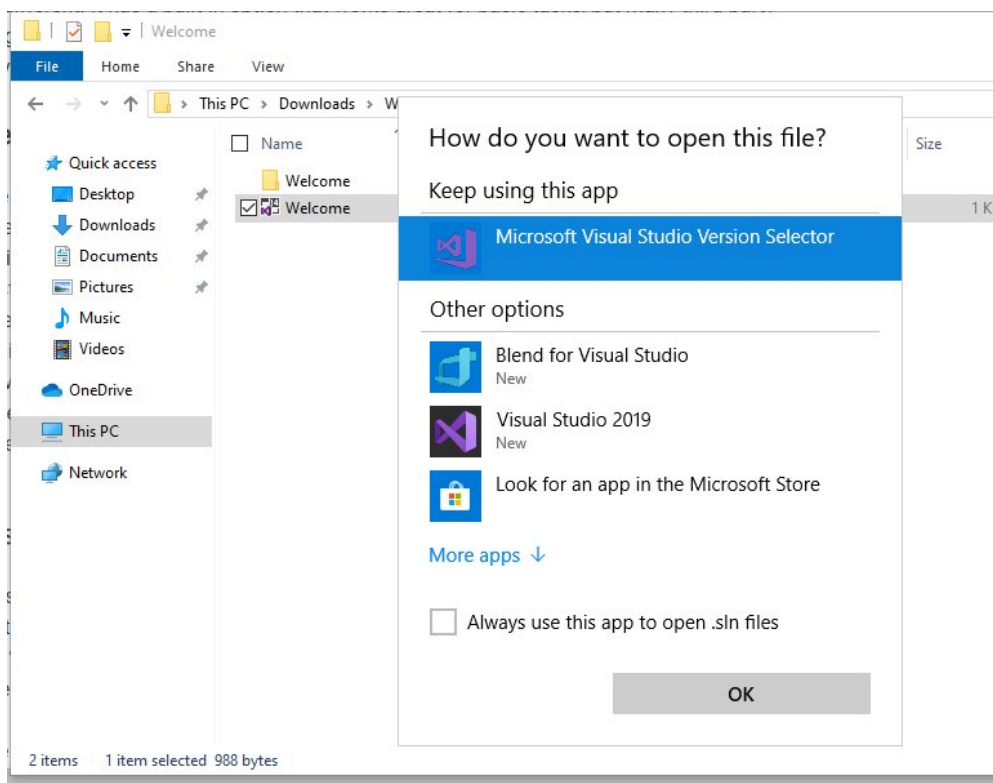
¹⁰<https://www.augusta.edu/its/computers-printing.php>

Note that none of them are endorsed by the school and that they can pose security and privacy challenges: never enter any sensitive information and do not rely on them too heavily. However, they can be a good support if you'd like to test a short snippet of code but don't have access at the moment to a computer with Visual Studio installed.

3 Your First Program

3.1 Opening Your First Program

1. Download [Welcome.zip](#) and save it on your computer.
2. Extract (Right-click on the file, then click on “Extract all”) this archive. **Do not simply double-click on it, as that would only give you a preview of the archive without actually extracting it.**
3. Go in the “Welcome” folder that was created.
4. Double click on the “Welcome.sln” file.
5. If you are prompted with a screen like this one:



Pick “Visual Studio 2019” (and **not** Visual Studio Code or Blend for Visual Studio).

6. Visual Studio (VS) should start, you don't have to register to the “Visual Studio Team Services Organizations” (but you can, using your @augusta.edu account, if you want), discard the security warning if there is one.
7. In the “Solution Explorer” to the right, expand all the items that can be expanded by clicking on the symbol.

3.2 Compiling and Executing Your First Program

1. In the Solution Explorer, double-click on `Program.cs`. This is the *source code* of the application you are actually considering.
2. Let's compile this program, using `Build` → `Build solution`. What happened?
3. Let's run this program, using `Debug` → `Start without Debugging`. What happened?

You will **extensively** compile and run programs in this class. Instead of having to click twice, I highly recommend that you start now memorizing shortcuts:

- Use `Ctrl + Shift + B` to build the solution.
- Use `Ctrl + F5` to start the program without debugging.

With `Alt + F4` (to exit any program), that makes 3 shortcuts already! You can find a complete list at <http://visualstudioshortcuts.com/>. I will try to introduce more useful shortcuts as we progress.

4 Backups

Now we need to make sure you know how to save your work and access it. This is especially important if you are using the computer lab rooms, as **you can not store files permanently on the lab's computer, you will have to store them either online in your cloud storage or on a usb stick.**

4.1 Finding The Right Tool

You can save your project:

- On your hard drive, if you are using your own computer.
- On an external/removable data storage: USB flash drive, external hard disk drive, or any kind of USB mass storage device.
- On a server: the University has a partnership¹¹ with `box.com`¹², and you can follow this tutorial¹³ to get started, but any service (Google Drive¹⁴, Dropbox¹⁵, OneDrive¹⁶, etc.) would do.

Having *two* backups is generally recommended.

If you chose the “virtual” option (i.e., using a server) and you are in a computer lab, **do not** install a synchronization software (like Google Drive and Sync¹⁷, Box's app¹⁸, etc.) on the lab computer: it will likely not work, due to University rules¹⁹. Instead, create the structure/project/files on the computer during the lab and upload them (using the web-interface) at the end of the lab. Make sure to always upload your files before logging out of the computer.

¹¹<https://www.augusta.edu/its/box/>

¹²<https://box.com/>

¹³<https://www.augusta.edu/its/box/quickstart.php>

¹⁴<https://www.google.com/drive/>

¹⁵<https://www.dropbox.com/>

¹⁶<https://onedrive.live.com/>

¹⁷<https://www.google.com/drive/download/>

¹⁸<https://app.box.com/services/browse/official>

¹⁹<https://augusta.policytech.com/dotNet/documents/?docid=5702>

4.2 Making Sure You Have the Right Files

Now that you know where to store your files, create a folder for this class, and a subfolder for the first lab. Your organization should look like the following:

```

└── csci1301
    └── 01_lab
        ├── Welcome.zip
        └── Welcome
            ├── Welcome.sln
            └── Welcome
                ├── Welcome.csproj
                ├── Properties
                │   └── AssemblyInfo.cs
                ├── Program.cs
                ├── obj
                │   └── Debug
                │       ├── Welcome.pdb
                │       ├── Welcome.exe
                │       ├── Welcome.csprojResolveAssemblyReference.cache
                │       ├── Welcome.csproj.FileListAbsolute.txt
                │       ├── TempPE
                │       └── TemporaryGeneratedFile_E7A71F73-0F8D-4B9B-B56E-
2E70B10BC5D3.cs
                │       ├── TemporaryGeneratedFile_5937a670-0e60-4077-877b-
f7221da3dda1.cs
                │       ├── TemporaryGeneratedFile_036C0B5B-1481-4323-8D20-
8F5ADCB23D92.cs
                │       └── DesignTimeResolveAssemblyReferencesInput.cache
                ├── bin
                │   └── Debug
                │       ├── Welcome.vshost.exe.manifest
                │       ├── Welcome.vshost.exe.config
                │       ├── Welcome.vshost.exe
                │       ├── Welcome.pdb
                │       ├── Welcome.exe.config
                │       └── Welcome.exe
                └── App.config

```

where `csci1301` and `01_lab` are folders, and `Welcome.zip` is a file.

You do not need to check that every single file and folder is here, just note that you have multiple folders, and that there are many files in the `Welcome` folder, not only the `.sln` and the `.cs`: make sure you copy the entire structure when you want to backup or share your program! In this case, copying the `Welcome` folder is enough.

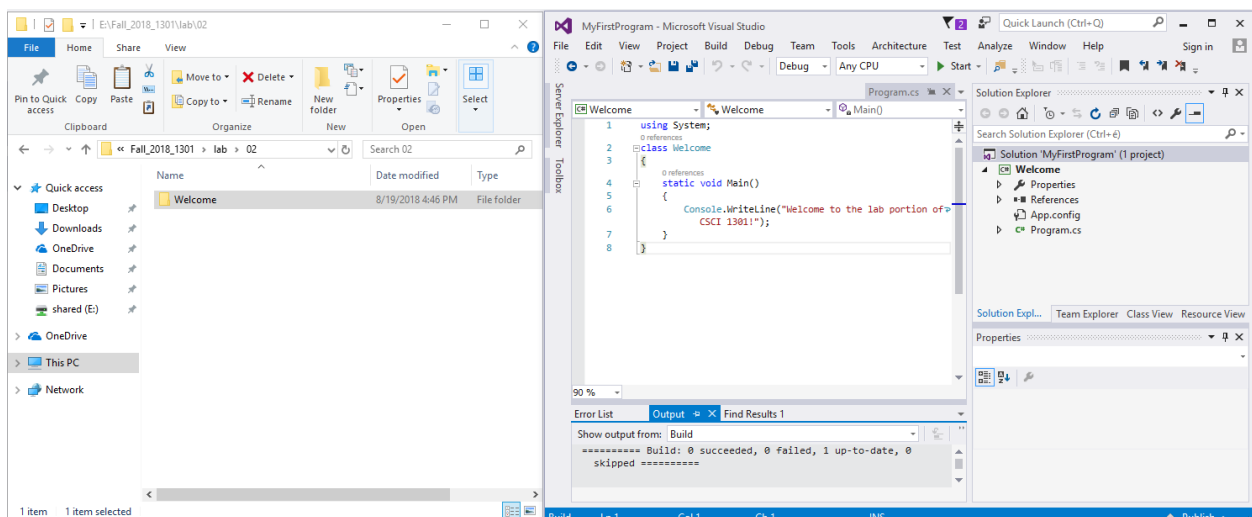
4.3 How Was the Backup?

Once you are done, test that you performed the backup properly. Re-download or transfer the files you just saved (the whole `Welcome` folder) on the computer and make sure you can still open the project with Visual Studio (VS). Do you remember how to build the solution and start the program without debugging? Using the shortcuts? If not, go have another look back at the “Compiling and Executing Your First Program” section.

If your backup went wrong (you can’t open the project, it won’t compile, ...), try to understand what happened. Then, re-download the “Welcome” archive²⁰, extract it, and make sure you can build the solution and start the program without debugging.

5 Orientation

Have a look at the following screenshot:



You should have roughly the same on your computer. If this image appears too small, you can zoom to have a clearer look.

Answer the following:

1. Where is the file explorer?
2. How many folders are in the folder opened in the file explorer?
3. How many files are in the folder opened in the file explorer?
4. Where is Visual Studio (VS)?
5. Where is the solution explorer?
6. Where is the window where you can edit the source code?
7. What is the output of the program you just built?

²⁰[Welcome.zip](#)

6 Breaking Your Program

If you followed the instructions carefully, you were able to build the solution and start the program without debugging after each step. As you know, C# has precise rules and not respecting them can prevent your solution from being built by VS.

In this exercise, you are asked to do the following:

1. Change the program so that it violates one of the syntax rules of C# (refer to your lecture notes and to the suggestions below).
2. Build the solution and note that an error is reported. In the build output (cf. the screenshot presented earlier, you may need to click on a tab in VS to see it), you will see a message like

```
===== Build: 0 succeeded, 1 failed, 0 up-to-date, 0 skipped =====
```

3. Click on the “Error list” tab next to the build output, and make sure you understand the error message.
4. Undo your change, using Ctrl + z.
5. Make sure you can build the solution without a new error message.

Do it three times, in order to identify three different error messages, and three ways of breaking C#'s rules.

If you have time or lack inspiration, you can try with the following, and see which one(s) make the building impossible (do not forget to undo your change after):

- Remove the semicolon after using `System`
- Replace `class Welcome` with `class TestOne`
- Remove the brace (or “curly bracket”, i.e., the `}` symbol) at the last line.
- Add three new lines at the end of the file
- Replace `Console.WriteLine` with `CONSOLE.WriteLine`
- Replace `Console.WriteLine` with `Console.WRITELINE`
- Add a new line between `Console.` and `WriteLine`
- Add a new line between `WriteLine` and `(`
- Add a new line between `Write` and `Line`
- Replace `Main()` with `Method()`
- Remove the indentation (i.e., the space between the beginning of the line and the first character of the instruction) on all lines