First Program

https://csci-1301.github.io/about#authors

May 24, 2021 (06:51:55 PM)

Contents

1 How Was the Backup?

If you are now in the computer lab on the lab computer, download on the computer the files you saved during the previous lab and make sure you can still open the project with Visual Studio (VS). Build the solution and start the program without debugging. Practicing this routinely will help you in making sure you know how to backup, transfer, and work on projects.

2 Re-using and Editing

Close Visual Studio (VS). With the file explorer, create a new folder (call it, for instance, `02_lab`) and copy the `Welcome` folder from the previous lab into it. In this exercise, you will rename and edit this copy.

2.1 Renaming

Open the copy of the `Welcome` folder that you just created with VS. Rename the solution **within** VS:

1. Right-click on "Solution 'Welcome' (1 project)" in the Solution Explorer, select "Rename", and rename it to "MyFirstProgram".
2. What change(s) do you notice in Visual Studio?
3. Can you still build and debug your program?
4. Look in the file explorer: did the name of the folder change? Did the name of the `.sln` file change?

You should *not* rename a solution with the file explorer: *always* use VS to rename.

2.2 Editing

We will now change (edit) our "MyFirstProgram" solution.

1. In `program.cs`, replace `"Welcome to the lab portion of CSCI 1301!"` with `"This is my first program."`.

2. Build the solution and start the program without debugging. Do you notice any change(s)?

3. Insert a new line after *// This is an in-line comment.* and before the first } sign, and paste the following:

   `Console.Write("This is my second message.");`

4. Build the solution and start the program without debugging. Do you notice any change(s)?

5. Insert another new line after the one you just created and paste the following:

   `Console.Write("This is my third message.");`

6. Build the solution and start the program without debugging. Can you notice the difference between `WriteLine` and `Write`?

7. Insert another new line after the one you just created and paste the following:

   `Console.Write("\t This is my fourth message.");`

8. Build the solution and start the program without debugging. Can you tell what `\t` is doing?

9. Insert another new line after the one you just created and paste the following:

   `Console.Write("\n This \n is \n my fifth message.\n");`

10. Build the solution and start the program without debugging. You should see something like this:

    Can you tell what `\n` is doing?

11. Have a look at the documentation page on escape sequences[1] and edit your program by adding a statement that displays the `\` and the `"` characters.

12. Add a comment (using *//* or */∗* and *∗/*) in your program.

Make a back up of what you just did: upload the `02_lab` folder you created previously on your remote backup or copy it on your thumb drive. **Do not** use the "Save as…" capacities of VS: close VS and copy / upload the folder "by hand" using the file explorer or a browser. Re-downloading or re-transfering it and re-opening it is a good way of making sure that your back up is correct.

3 Your First New Project

This time you will not be given a project to load or to copy. You will start "from scratch". Start by creating a folder for this lab.

---

[1]https://docs.microsoft.com/en-us/cpp/c-language/escape-sequences

3.1 Starting from a template

We will first create a new project for Visual C# using the template for "Console App (.NET Framework)" or "(.NET Core)". The difference between ".NET Framework" and ".NET Core" does not matter for this class.

1. Launch Visual Studio.

2. Create a new project using Ctrl + Shift + N or "File" → "New" → "Project"

3. Find the "Console Application Visual C#" (a.k.a. "Console App (.NET Framework) Visual C#" or "(.NET Core)"; it does not make a difference for this class.) template by using Ctrl + E and then typing "Console", or by navigating using the menu on the left panel: "Templates" → "Installed" → "Visual C#" → "Windows" → "Classic Desktop" (or "Installed" → "Visual C#" → "Classic Desktop"). If you can't find it, follow the instructions in the note at https://docs.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-console?view=vs-2019 to install it. Complete the next step before effectively creating the project.

4. Enter "MyFirstProject" as the name of the project, and enter "MyFirstSolution" as the name of the solution. For the Location, choose a good place to save your solution; the best pick would be a folder you created for this lab.

5. Leave the rest as is and click on "Create" or "Ok".

6. Now answer the following:

    a) A source code file appeared in the main window of VS. Compare this code with the code you studied previously: How are they different? How are they the same?
    b) Open a file explorer and navigate to the folder where you stored your project (i.e. the folder you chose in the "Location" field when you created it). Open the project's folder and compare it with the folder for the `Welcome` project: How are they different? How are they the same?
    c) Try to compile this code using Ctrl + Shift + B or `Build` → `Build solution`. Did the compilation succeed?
    d) Execute the code using Ctrl + F5 or `Debug` → `Start without Debugging`. What happened? Compare with what happened with the `Welcome` project.

3.2 Editing the template

Now you will start writing your own code. We'll start by writing a very familiar instruction to display a message on the screen.

1. Place the cursor inside the `Main` method, after `static void Main(string[] args)` and the opening brace (i.e.`{`). Create a new line.

2. Type `Console`. The (at first sight annoying) auto-completion feature that displays suggestions and messages as soon as you start typing is called Intellisense. You can read about it at https://docs.microsoft.com/en-us/visualstudio/ide/using-intellisense. You'll probably end up using it a lot, but let's not worry about that for now.

3. Type in `.Wri` after `Console` (don't forget the period!) and notice that Intellisense is already making good suggestions: you actually want to write `WriteLine`! Either finish writing `WriteLine` or select it from the menu that appears.

4. Now type an open parenthesis, i.e. `(`, and notice that Intellisense closed it for you and is already displaying another message.

5. Type the string of your choice between those two parenthesis, i.e. something like `"This is my first message"` (and don't forget the quotes).

   At this point, your `Main` method should look like this:

```csharp
static void Main(string[] args)
{
    Console.WriteLine("This is my first message!")
}
```

6. Compile (build) your file. Oh no, something went wrong! Can you fix this problem?

7. Once you can compile your program without errors, execute (run without debugging) it.

8. Make a backup of your project.

4 Exploring the Documentation

The documentation for Visual Studio and C# is packed with useful information and efforts are made to make it accessible to beginners. The goal of this exercise is to help you realize that it contains answers to questions you may have asked yourself, like "what is a solution?" or "what does the `namespace` keyword do?".

All the documentation for Visual Studio is at https://docs.microsoft.com/en-us/visualstudio/. The documentation for C# is at https://docs.microsoft.com/en-us/dotnet/csharp/. To get started, have a look at the first three paragraphs of https://docs.microsoft.com/en-us/visualstudio/ide/creating-solutions-and-pr ojects (or at https://docs.microsoft.com/en-us/visualstudio/mac/projects-and-solutions?view=vsmac-2019 for Mac) and answer the following:

1. What is a solution?
2. What is a project?
3. Which one contains the other: the solution or the project?

Note that since we will only manipulate one project at a time, the distinction between solution and project will not be critical in this lab. That's the reason why I wrote previously "create a new project" with the sense of "create a new solution and a new project in it": we will never manipulate more than one project in a solution at a time.

Finally, read the page at https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/namespaces /. Do you know an example of a namespace that we have used?

5 More About Displaying Characters on the Screen

1. Create a new project.
2. Edit the `Main` method so that when compiled and executed, your program will display the following on the screen:

```
1     !
2    !!!
3  !!!!!!!
4
5  Press any key to continue . . .
```

1. We will now use the "Find and Replace" feature of VS[2].

---

[2]https://docs.microsoft.com/en-us/visualstudio/ide/finding-and-replacing-text?view=vs-2019

a) Hit "Edit" → "Find and Replace" → "Find in Files", or press Ctrl + Shift + F.

b) In the panel that appears, click on the "Replace in Files" tab, enter ! under "Find what:", and * under "Replace with:".

c) Hit "Replace All" and note the modifications in your program.

2. As you can see, this is a really useful feature of VS, but also a really dangerous one. If you were to replace all the "*" characters with "!" in all the programs we wrote so far, what could possibly go wrong?