CSCI 1301 – Lab 03

January 15, 2019

1 How Was the Backup (bis)?

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

2 Your First Own Project

This time, you will not be given a project to load, but you will start "from scratch". Start by creating a folder for this third lab.

2.1 Starting from a template

We will first create a new project for Visual C# using the template for "Console App (.NET Framework)".

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#") 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"). Complete the next step before effectively creating the project.
4. Enter "MyFirstProject" as the name of the project, select the right location (the folder you created for this third lab), and enter "MyFirstSolution" as the name of the solution.
5. Leave the rest as it is, and click on "Ok".
6. Now, answer the following:
   a) A source code 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. 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 was happening with the `Welcome` project.

2.2 Editing the template

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

1. Place the cursor inside the `Main` method (i.e., after the brace after `static void Main(string[] args)`). Create a new line.
2. Type `Console.` The (at first sight annoying) auto-completion feature that display suggestions and messages as soon as you start typing is called Intellisense. You can read about it at https://msdn.microsoft.com/en-us/library/hcw1s69b.aspx or https://docs.microsoft.com/en-us/visualstudio/ide/using-intellisense, you'll probably ending 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 appeared.
4. Now, open a parenthesis, i.e., type `(`, 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:

   ```
   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 error, execute (= "run without debugging") it.
8. Make a back-up of your project.

3 First Variables Manipulation

1. Create a new project as indicated above. Save it in the same folder as the previous one (that is, in a special folder for the third lab), and give it a different name (like "VariableManipulation").
2. In the `Main` method, add three statements:
   - One that declares a `string` variable named `lastName`,
   - One that declares a `string` variable named `firstName`,
   - One that declares an `int` variable named `classOf`
3. Below those three statements, write statements that assign your last name to the first variable, your first name to the second variable, and your anticipated graduation year (i.e., 2025, for instance) to the third one.
4. Print the values of the three variables, using the following statement (that uses interpolation):

   ```
   Console.WriteLine($"My name is {firstName} {lastName}, and I expect to graduate in
   ↪ {classOf}.");
   ```

5. Compile and execute your program. It should display a message like (with your own name and graduation date, of course):

   ```
   My name is Clément Aubert, and I expect to graduate in 2025.
   ```

   Fix it if it is not the case.

6. Answer the following by first thinking of what may happen, and then editing your program to check your hypothesis. Whould the compilation still be successful if we were to try to…

   a) … assign a value to a variable before declaring it?
   b) … display the content of a variable before assigning a value to it?
   c) … assign a `String` value to an `int` variable?
   d) … assign an `int` value to a `String` variable?

7. At the end of your main methode, add three statements that change the value of the three variables, and copy the `Console.WriteLine` statement that was previously given. Notice that the very same statement will now print a different message!