Your First Own Program

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

May 27, 2021 (09:25:27 PM)

Contents

In this lab you will first edit an existing program then create a new program "from scratch". You will also explore C# documentation and learn new useful IDE features.

1 Your First Own Program

1.1 Editing Existing Program

1.1.1 Re-using and Editing

1. Download InitialSolution.zip and save it on your computer.
2. Unzip the program file[1]
   - on Windows: right click and choose "Extract all"
   - on macOS: double-click the `.zip` file
   - on Linux: right click and choose "Extract" or "Open with Ark"[2]

1.1.2 Renaming

Open `InitialSolution` that you just extracted in your IDE.

[1] Be careful: some file explorers will simply *preview* the (zip) archive if you simply double-click on it, but most IDE will not accept to open a file if it has not been actually extracted / unziped!

[2] https://www.wikihow.com/Unzip-Files-in-Linux

1.1.2.1 Renaming the Solution

Use your IDE to rename this *solution*:

1. Right-click on `InitialSolution` (in "Solution Explorer", "Explorer", or even "Solution").
2. Select "Rename" (it can be under "Edit"), then rename the solution to `EditedSolution`.
3. What change(s) do you notice in your IDE?
4. Can you still build and debug your program?
5. Look in your file system where you unzipped the solution earlier. Did the name of the project directory change? Did the name of the `.sln` file change?

***Do not* rename a solution outside your IDE: always use an IDE to rename.**

Renaming C# project files requires more than simply changing a file name. By using an IDE to perform renaming, all references to the name will be updated.

1.1.2.2 Renaming the Project

Next use your IDE to rename the *project*:

1. Right-click on `InitialProject`.
2. Select "Rename" then rename the solution to `EditedProject`.
3. What change(s) do you notice in your IDE?
4. Can you still build and debug your program?
5. Look in your file system where you unzipped the solution earlier. Did the name of the project directory change? Did the name of the `.csproj` file change?

1.1.3 Editing

We will now change (edit) our `EditedSolution` solution.

1. In `Program.cs`:
   - replace `"Welcome to the lab portion of CSCI 1301!"`
   - with `"This is my first program."`.
2. Build and run the program. Do you notice any change(s)?
3. Insert a new line after the existing `Console.WriteLine` and before the first `}` sign and paste the following:

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

4. Build and run the program. 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 and run the program. 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 and run the program. 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 and run the program. You should see something like this:

```
This is my first program.
This is my second message.This is my third message.    This is my fourth message.
 This
 is
 my fifth message.
```

Can you tell what `\n` is doing?

11. Have a look at escape sequences[3] 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 `EditedSolution` to your remote backup or copy it on your thumb drive. After saving the backup, close your IDE and make sure you can still open the solution. Re-downloading and re-opening solutions is a good way of making sure that your backup is correct.

1.2 Creating Your First New Project

This time you will not be given a project to load or to copy. You will start from scratch. If your IDE is currently open, exit your IDE application. Next create a new folder for this lab in your backup directory.

1.2.1 Starting from a Template

We will first create a new C# project using the template for a "Console App".

1. Launch your IDE
2. After the IDE launches, look for an option to create a new project. The exact wording varies between different IDEs, but look for one of the following:
   - `Create a new project` or `New` or `New Solution` in the launch screen
   - `File > New > Project` in the IDE menu
3. Look for "Console Application" option and check that the associated language is C#.
   - *note to Windows users*: you may see multiple options for a framework (.Net Framework or .NET Core) and can choose either, it does not make a difference for this class.
4. Enter `MyFirstProject` as the name of the project.
5. Enter `MyFirstSolution` as the name of the solution.
6. For location or solution directory: choose a good place to save your solution. The best pick would be a folder you created for this lab.
7. Leave the rest as is and click on "Create" or "Ok".
8. Now answer the following:

   a) A source code file appeared in the main window of your IDE. Compare this code with the code you studied previously: How are they different? How are they the same?

   b) In your file system, navigate to the directory where you stored your project. Open the project directory and compare `MyFirstSolution` with `EditedSolution` project you worked on earlier. How are they different? How are they the same?

[3] /book.html#escape-sequences

c) Try to compile `MyFirstSolution`. Did the compilation succeed?

d) Execute `MyFirstSolution`. What happened? Compare what happened with the `EditedSolution` project.

1.2.2 Editing the Template

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

1. Place the cursor inside the `Main` method, after `static void Main(string[] args)` and the opening brace (i.e.{).
2. Create a new line.
3. Type `Console` then pause
4. After a short moment, an auto-completion feature that displays suggestions and messages should display. This is a common IDE feature to help the programmer. You'll probably end up using it a lot, but let's not worry about it for now.
5. Type in `.Wri` after `Console` (don't forget the period!) and notice the good suggestions: you actually want to write `WriteLine`! Either finish writing `WriteLine` or select it from the menu that appears.
6. Now type an open parenthesis, i.e. `(`, and notice that it is closed for you automatically.
7. Type a `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!")
   }
   ```

8. Compile (build) your program.
9. Oh no, something went wrong! Can you fix this problem?
10. Once you can compile your program without errors, execute it.
11. Make a backup of your project.

1.3 Exploring the Documentation

The documentation for 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?".

The documentation for C# is at https://docs.microsoft.com/en-us/dotnet/csharp/. To get started, have a look at "Introduction" at https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/language-specification/introduction/ and answer the following:

1. What C# langauge feature is responsible for reclaiming unused memory?
2. What file extension is used by C# source code files?
3. Can you list 3 different C# data types?

C# programs often use namespaces as a way of organizing large code projects, and your IDE may create a `namespace` when you create a new program. 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?

1.4 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. We will now use the "Find and Replace" feature of your IDE. Look for it in the top menu of your IDE, typically `Edit > Find and Replace > Find in Files` or `Edit > Find > Replace`.
2. In the panel that appears, enter the following:
   - (Find what) `!`
   - (Replace with) `*`
3. Hit "Replace All" and note the modifications in your program.
4. As you can see, this is a really useful feature of your IDE, 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?