# PE1 - Hello World

## Objective

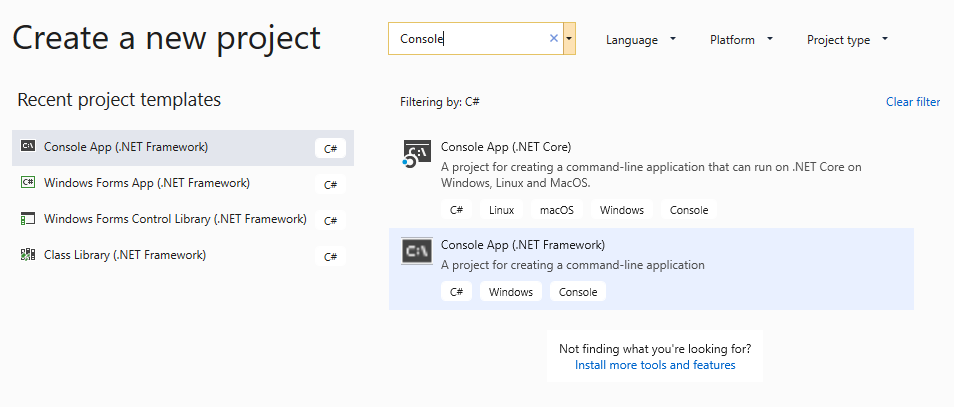
To get familiar with Visual Studio 2019, the tool that lets you create your own C# programs.

This exercise should function as a guide for creating console C# programs throughout this semester.

## Opening Visual Studio & Project Setup

1. Go to the start menu and look for Visual Studio 2019. While the start menu is open, you can begin typing “Visual Studio 2019” to find it quickly.
2. If this is the first time Visual Studio is being run on this computer, it may ask you to choose “Default Environment Settings”. Choose “Visual C#”. You can always reset this choice later if you need to.
3. Once Visual Studio has opened, you’ll need to **create a new C# Project**. A Project contains any and all code for a program, in addition to settings and other information. All code you write in Visual Studio must be part of a Project.
4. To create a project, either:
   * **Click “Create a New Project…”** on the start page that opens when Visual Studio is opened, OR
   * Go to **File > New > Project…** This will open the “New Project” window.
5. There are a few choices you’ll need to make next.
   * The first step is to make sure the correct programming language is chosen in the left panel. Visual C# will either be selected by default (if it’s the default environment), or it’ll be listed somewhere in the left panel or under “Other Languages”. **Find and click on “Visual C#”.**

This will present you with all of the C# Windows projects you can make in the center panel. Type "Console" in the search box to filter. **Click on “Console App (.NET Framework)” and click Next.**

1. 
   * Always replace the *Project Name* default with something more meaningful (the name of the practice exercise, exam or homework you’re working on). “Lastname\_PE1” or “Lastname\_HelloWorld” would work well.
     1. This name will be used in several places, including the name of the folder that will be created automatically.
     2. For now, **use “Lastname\_HelloWorld” for the *Name*.** **Don’t click OK yet!**
   * Navigate to your Desktop for a moment. Create an empty folder named IGME 201.
   * Inside *Location,* click the "…" Browsecontrol**, find and select the Desktop, find IGME 201, then click “Select Folder”**. This will create a new folder in your IGME 201 desktop folder, called “HelloWorld”, in which you’ll find all of the files you’ll be working with today.
     1. You may find it useful to use this IGME 201 folder to store all of the exercises and assignments for this class or on a USB drive.
   * For *Solution,* “Create new solution” should appear in the drop-down by default.
   * Leave "Place solution and project in same directory" **checked**.
2. After pressing OK, Visual Studio will create several files and folders for you, and open the most important one: This file is called “Program.cs”, and it is where you can start writing code.

## Code

Now that you’ve got Program.cs open, let’s look at the auto-generated code.

**Using Statements**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

These statements tell the program which parts of the C# language (called namespaces) you might be using in your program.

Since this is going to be a simple program, you’ll only need the first one ( using System; ). The last four are included because they’re commonly used, but they won’t be necessary for a while. You can leave them there, or feel free to remove the last four if you’d like.

**Namespace**

namespace HelloWorld

{

Namespaces help organize your code, and can help prevent naming conflicts. All code in the same namespace can be referenced without the need for more “using” statements.

The default namespace has the same name as the project. In smaller programs (the kind you’ll make in this class), all of your code will generally be in the same namespace.

The start and end curly braces – these things: { } – define the start and end of a section (or “block”) of code. In this case, they define what is in the “HelloWorld” namespace.

**Class**

class Program

{

All C# programs need at least one class. Most of your code will be inside of classes. Notice that the class has its own set of curly braces, which define the start and end of the class.

**The Main Method**

static void Main(string[] args)

{

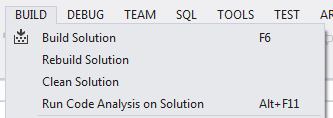
}

Every program needs a place to start. In C#, like many other languages, the starting point is called “Main”. There’s yet another set of curly braces defining exactly what is contained in the Main method (which, right now, is nothing).

## Compiling (a.k.a. Building) Your Program

Before you can run your program, you have to compile it. This is also known as “building” in C#. When you build your code, Visual Studio creates a standard Windows executable file. You can also have it build the code and run the program in one step.

1. By default, the shortcut for Building is F6. It may be different on your machine depending on the default language setting in Visual Studio. Either way, you can also build by going to the BUILD menu item and selecting “Build Solution” (which will show you the current keyboard shortcut).

**What is the keyboard shortcut for “Build Solution” on *your* computer? (It may or may not be F6.)**

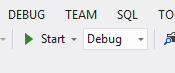
**Build your program now**, by pressing F6 or by using the menu. You should see the status bar at the very bottom of the Visual Studio say “Build succeeded”. Each time you build, Visual Studio saves all of your files automatically.

If there are errors in your code, the build may fail. In that case, Visual Studio will display a list of the errors that it found. We’ll look at common errors and how to fix them as the course progresses.

Building your program will not actually show your program running in the console window. Let’s do that next.

## Running Your Program

1. The next step is to actually test your program by running it. There are a few ways to do this. The first is to press the green “Start” button on the toolbar.



The other option is to go to the DEBUG menu, and choose “Start Debugging”. You can also press F5 (the default shortcut for starting the program).

**What is the keyboard shortcut for “Start Debugging” on *your* computer? (It may or may not be F5.)**

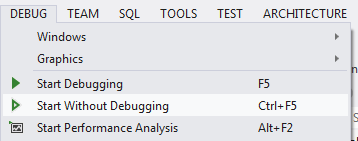
**Run your program now**, either by clicking the button, the menu or the keyboard shortcut.

1. You should notice a small window pop up and then immediately disappear. That was your program running. Visual Studio built an executable file (called HelloWorld.exe) and ran it for you.

**Why do you think the program closed?**

1. Having the program close automatically isn’t helpful if we want to see some output. We’ll need a way to keep the window open so we can tell if our code actually worked. There are several ways to do this, and we’ll discuss the pros and cons of each as we cover them.

For now, you’re going to use a slightly different command to run the program and keep it open. You can press Ctrl+F5, or go to the DEBUG menu and choose “Start without Debugging” to run the program. The side effect of running the program this way is that the window will remain open when the program finishes.



**What is the shortcut for “Start without Debugging” on *your* computer?**

**Run your program again**, this time getting the window to stay open. If you have any problems, ask the instructor or the TA for help.

1. If you were successful, you should see a mostly blank window with the words “Press any key to continue…” Visual Studio adds those words to let you know the program has finished (they’re not actually part of your program).

Now that you can keep the window open, we should make it do something slightly more interesting.

1. We’re going to add a statement that prints some text to the window (also known as the “console”). Since the starting point for the program is the Main method, you’ll need to add a line of code between the curly braces that come after the Main method.

You don’t need to edit the line that has the word “Main”. Simply **add a blank line between the two curly braces, and type the following**:

Console.WriteLine(“Hello World!”);

You’ll need to match capitalization, and remember to put a semicolon at the end. When you’re done, your Main method should look like this:

static void Main(string[] args)

{

Console.WriteLine("Hello World!");

}

As you type, you may notice small menus popping up. This is called Intellisense, and it is Visual Studio’s way of showing you what code you can use in different situations.

1. Now you can test your program again. **Build and run your program**, using the same commands you did previously. Remember to get the window to stay open.

If you have any errors or problems, ask the instructor or the TA for help. Don’t move on until you can run your program, keep the window open and see the words “Hello World!”

## Changing Your Program

1. Let’s change things up a bit. Instead of printing “Hello World!” change your program to print your name.

You’ll need to **replace the “Hello World!” portion** of the line of code you wrote with something else. Remember to keep the quotes, as you are printing a string of characters! As an example, your code might look like this now:

Console.WriteLine("Jada Pinkett Smith ");

**Run the program again** to verify your changes worked. Super fancy, I know.

1. Obviously printing just your name is easy. Play around with the syntax you know from JavaScript to see if and how the syntax differs. Try some or all of the following:
   * Create some variables
   * Do some math calculations
   * Print the results
   * See if implicit and explicit casting works the same way
   * Try out if statements, for loops, while loops, etc.

## Submission

Add, Commit and Push the solution to your GitHub repository.

Submit the URL of your GitHub project to the assignment in myCourses.