# In-Class Worksheet #1

You should have Visual Studio 2008 open.

1. Create a new Project either from the shortcut on the Start page or through the file menu,
2. Select the type to be a C# Console Project
3. In the Location textbox, enter your desired hard drive location. Note, that this will be on the mounted z: drive in the CSE environment.
4. Select OK.
5. The Program.cs file should have been created and opened automatically.
6. Take some time to familiarize yourself with the IDE. I usually work with the Solution View (rather than the Class view) to the right, with the Properties View underneath it.
7. Clean-up some of the automated code:
    a. Remove all of the using statements from the top of Program.cs
    b. In the Solution Explorer (View->Solution Explorer), expand the Properties and the References tree views.
    c. In the References, select everything but System and System.Core. Right-click and select Remove.
8. Configure the IDE:
    a. Open the Options dialogue (Tools->Options).
    b. Under Fonts and Colors, change the Text Editor Font Size to your liking (I'm old so I use a bigger font).
    c. Under Help, Change the Search results per page to 50.
    d. Expand Help, under Online, uncheck Questions. In Codezone uncheck sites you do not want to see content from (I only use C# Corner, CodeGuru and DevX).
    e. Scroll down to Text Editor and expand it to find the C# options. Under General, check the box to show line numbers.
    f. Explore the other options available and click OK when done.
9. In Program.cs, position your cursor after the { in Main and hit Enter. Note that things are automatically indented for you.
10. Make Hello World:
    a. Type a capital S. Note the intellisense that is highlighting System. Now type a period ('.').
    b. Type Co and then use the arrow keys to scroll down to Console. Hit return and then a period (or simply hit a period again).
    c. Type WriteLine(. Note that intellisense now gives you 19 method signatures to follow.
    d. Click somewhere else and then click back to after the (. You can get intellisense back by deleting the paren and retyping it or, by hitting Ctrl->Shift->space. Use the arrows (either arrow keys or click on the arrow icons with the mouse) to reveal the syntax for a string.
    e. Type in "Hello C# World");
    f. Hit enter and type in garbage++. Note that the Error List window (View Error List) is displaying an error indicating a missing semicolon. Go ahead and add a semicolon to the end of the previous line.
11. Build your application
    a. Select Build->Build Solution
    b. The Error list pane has colored buttons at the top of it. These are toggle buttons. You can turn on the display of Warnings, etc. Here we have one Error. Toggling the error button hides it (not something we want to do).
    c. Delete the line of text garbage++;
    d. Also note the color coding on the side of the text window. Yellow indicates changed since last build, green indicates changed during this session. Deletions are not shown.

e. Rebuild the solution.

12. Debug and Run your Hello World:
    a. Click Debug->Start Debugging. Either:
        i. Your program just ran, finished and cleaned up. How rude, it did not even say hello! You also saw a bunch of windows flash.
        ii. You received a Security Exception. This is due to the fact that .NET by default does not want to trust applications that are not installed. More specifically, applications that try to run from a mounted file system. This issue should be resolved on the CL 112D machines. If not, send an e-mail to help. A work around is to copy the executable over to a local temporary directory and execute it. If need be, I will go over how to this within Visual Studio automatically in class.
    b. At line 8 (if you followed along exactly, this should be the line under the Console.WriteLine statement), click in the grey area to the left of the 8. This will place a ***breakpoint*** at this location.
    c. Re-run your program again with the debugger. Note that it now stops at line 8 (it is highlighted). You also have many debugger windows and your IDE layout has changed.
    d. Move your mouse over the args parameter to the Main function. Intellisense will indicate that it is an array of strings of size zero. Occasionaly, you will see a magnifying glass here. These pull up data specific visualizers (e.g., text will open a text window).
    e. Under the Debug menu the first item is *Windows*. This is a little confusing, but these are all the windows you open when debugging an application. Look at the Locals window. The only variable is args. This is an alternative (and the main way) to examine the variables.
    f. You should also see a new process is running in the taskbar. This is the Console window. The application is now stopped and you can look at the Console window and see your message.
    g. Make sure the Debug toolbar is visible. View->Toolbars->Debug. Click the Step Over tool button. This will normally execute the current line and stop at the next, but since this is the last line of our simple program, it runs the application until completion.
13. Final Thoughts
    a. If you look at the files created, the key ones are the Program.cs, Properties/AssemblyInfo.cs, your solution file (.sln) and your project file (.csproj).
    b. When you created this project, it asked if you wanted to put the solution into a separate directory. Visual Studio can have many projects within a solution, including several dll projects and even several executable projects. With the Pro version of Visual Studio, these can be C++, C#, Visual Basic and any combination of such projects. The .sln file is just a container that points to the various .proj files.
    c. Both the .sln and the .csproj files are XML files. Open them in notepad to get a feel for what goes in there, but in general, do not change them.
    d. The .suo file contains Solution User Options. This includes which files were open last time you closed it, windows opened and other IDE specific settings. It also indicates which project is the “StartUp Project”.
    e. You will notice in the Solution Explorer that your project ConsoleApplication1 is in bold. This indicates that it is the StartUp project. This is the project that Visual Studio will try to execute when you run the debugger or start without debugging. If you right click on this you will see options for the project, including to set this as the StartUp project. Since it is the only one in this simple example it selected it automatically.
    f. Select Debug->Start Without Debugging. Note that the Console window displays and prompts you to hit a key to continue.
    g. Finally, you can have several builds for the same project, including a Release version that optimizes the code more and strips the debugging information. You can also build other versions as your needs dictate.