|  |  |
| --- | --- |
| **GAT150 – INTRODUCTION TO GAME PROGRAMING** |  |

**PROJECT SETUP**

***OVERVIEW***

In this assignment, you will install Visual Studio, create the solution/project and then add the project to a source control repository. Visual Studio is known as an IDE (Integrated Development Environment). An IDE is a program that contains a set of tools (editor, compiler and debugger) to create software programs. Version control, also known as source control, are systems responsible for the management of changes to documents like source code. Git is a popular Version Control System (VCS) and is the VCS we will use for managing our source code revisions (changes). GitHub and Bitbucket are both Version Control hosting sites that can store the Git.

In the second part of the assignment, the SDL (Simple DirectMedia Layer) library will be added to the project. The SDL library adds additional graphics and input functionality to our program.

***GRADING***

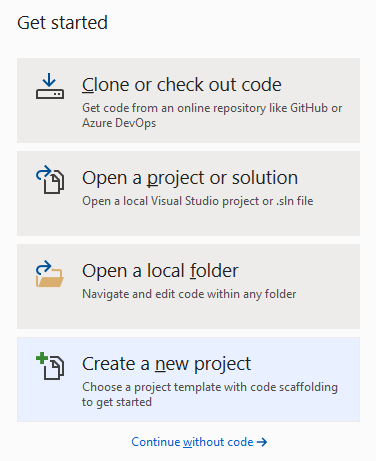
This assignment is worth 50 points. To receive full credit for this assignment, you need to show Visual Studio with an SDL window displayed.



***INSTRUCTIONS***

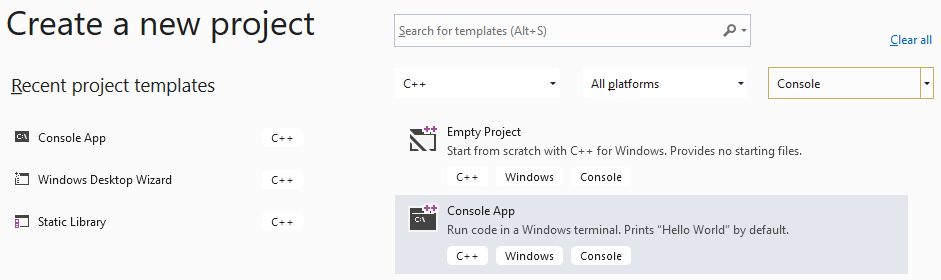
# Create Visual Studio Project

* Open *Visual Studio* and *Create a new project*

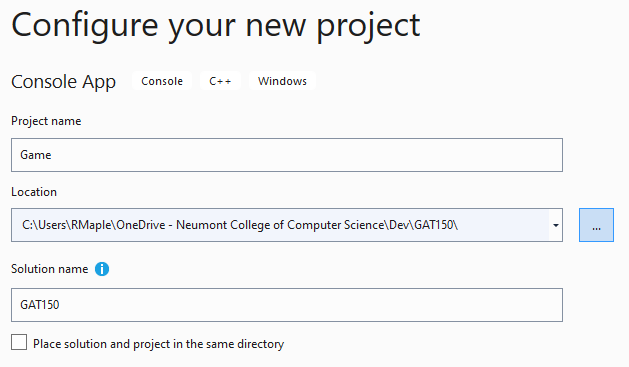


* Create a new *Console App* project
  + To reduce project types listed, set *Language* to *C++* and *Project type* to *Console*

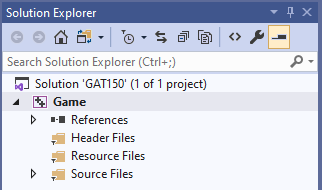




* Name your project and solution
  + the *Project name* is ‘*Game’*
  + the *Solution name* is *‘GAT150’*
* Set the location to the directory you want to keep your Solution

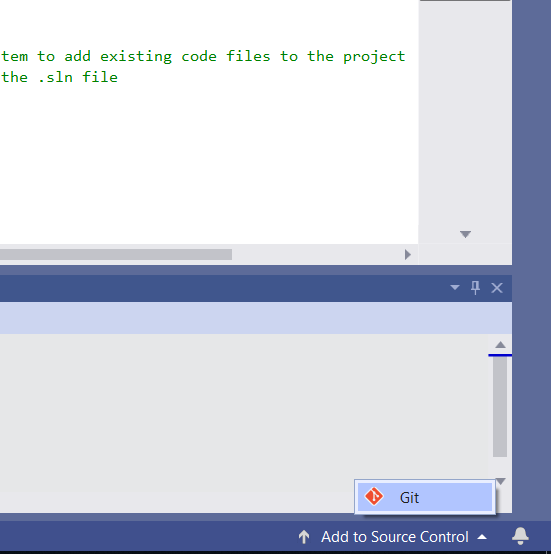


* Create the project, the *Solution Explorer* should look like this image

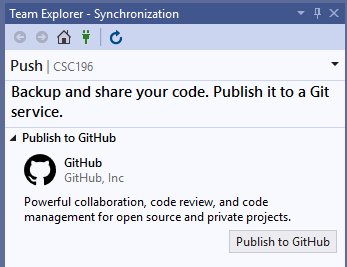
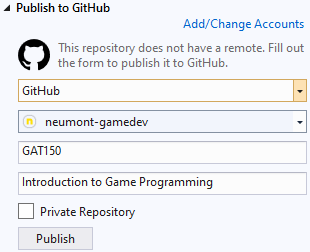


# Add the project to Source Control

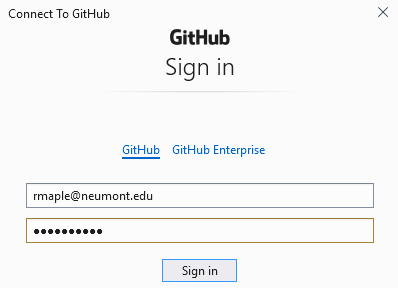
* Click *Add to Source Control* located at the bottom right of the *Visual Studio* IDE
* Select *Git*



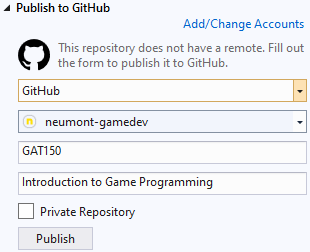
* Publish the project to *GitHub*

* *Sign in* using your GitHub email and password or *Sign in with your browser*
  + This step may not be needed

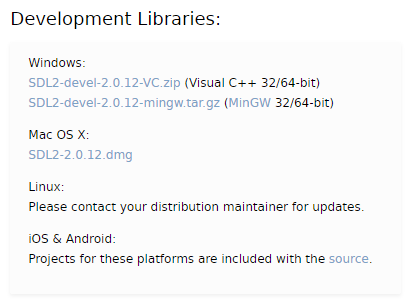
* If successful, the GitHub account will appear and the *Publish* button will be active
  + Click the *Publish* button



# Download SDL files

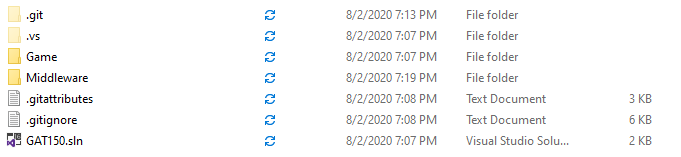
SDL - “Simple DirectMedia Layer is a cross-platform development library designed to provide low level access to audio, keyboard, mouse, joystick, and graphics hardware via OpenGL and Direct3D.”

* Download the SDL library
  + <https://www.libsdl.org/download-2.0.php>
  + <https://www.libsdl.org/release/SDL2-devel-2.0.12-VC.zip>

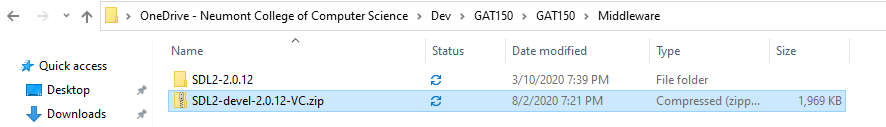


**Middleware** is software that provides common services and capabilities to applications outside of what's offered by the operating system.

* Create a folder called Middleware in your solution folder

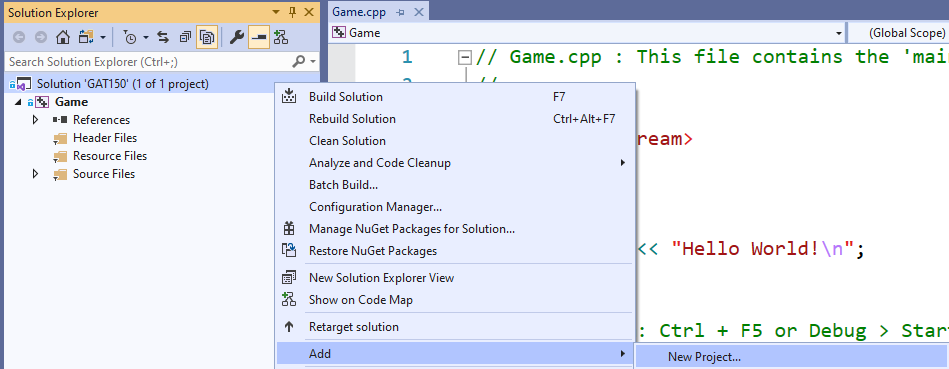


* Place the downloaded .zip file in the Middleware folder
* Extract the contents of the .zip file in the Middleware folder
  + You can delete the .zip file after extracting

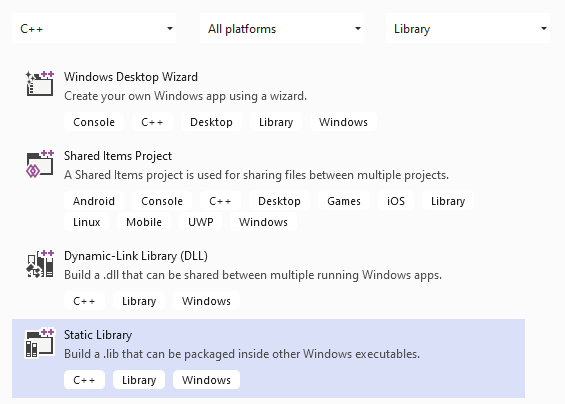


# Create an Engine **Static Library** project

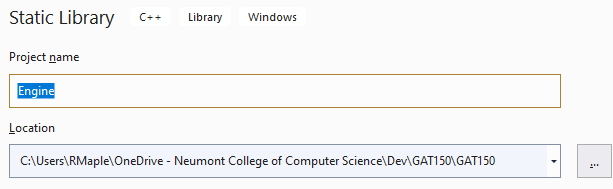
* In the Solution Explorer right click the GAT150 Solution
  + *Add > New Project…*



* Create a *Static Library* (Select *Library* as *Project Type* if not seen)



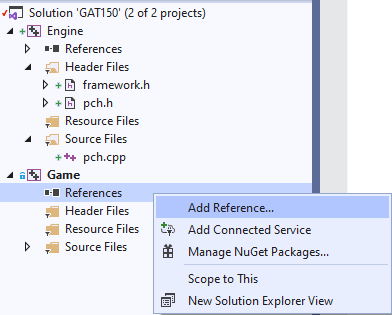
* Name you Static Library ‘Engine’
  + Create



* Delete the Engine.cpp file (it is not needed)
  + Select file and press *Delete*



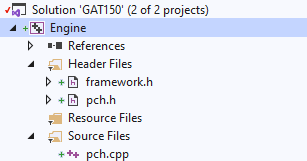
* Add the Engine Reference to the Game project
  + Right-click *References* in the *Game* project and *Add Reference…*
  + Select *Engine* and click *OK*



# Add SDL includes to the Engine project

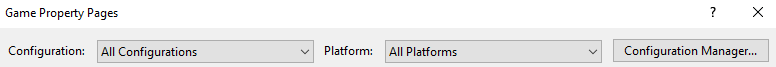
* Include the directory that the *SDL* includes (.h) are located
  + This will allow the program to find the SDL include files (.h)
  + Right-click the *Engine* project



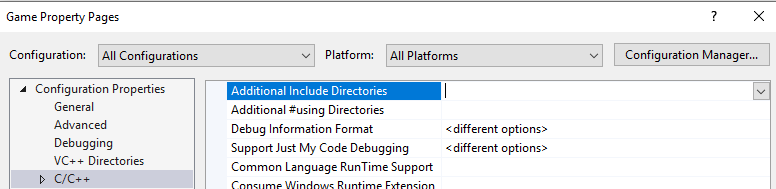
* + Go to the bottom of the drop down to *Properties*



* **Make sure that *Configuration* is set to *All Configurations* and *Platform* is set to *All Platforms***
  + This will make our changes apply to both x86/x64 and Debug/Release builds



* Go to the C/C++ properties and select the drop-down arrow in *Additional Include Directories*



* Select *<Edit…>*

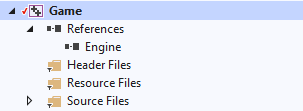


* Add the include directory, this will be a *Relative* directory from the Solution Directory
  + *$(SolutionDir)Middleware\SDL2-2.0.12\include*
  + This is done by using the $(SolutionDir) macro which will start our directory location from where our solution is located
    - *Relative* means that our directory *Middleware\SDL2-2.0.12\include* is located relative to the Solution Directory



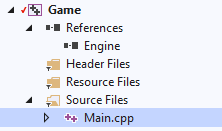
* Click *OK* and *Apply* to save this change and exit *Properties*

**Repeat the same process for the *Game* project so files in the *Game* project can see the SDL includes**



# Add code to initialize the SDL

* Rename the Game.cpp file to Main.cpp



* Replace the code in Main.cpp with the following:

#include <iostream>

#include <SDL.h>

int main(int, char\*\*)

{

if (SDL\_Init(SDL\_INIT\_VIDEO) != 0)

{

std::cout << "SDL\_Init Error: " << SDL\_GetError() << std::endl;

return 1;

}

SDL\_Quit();

return 0;

}

* This will initialize the video system for SDL
* Build and run the program
* You’ll get errors about *unresolved external symbols*
  + This means there are no function SDL definitions for the declarations in the SDL headers

2>Main.cpp

2>Main.obj : error LNK2019: unresolved external symbol \_SDL\_GetError referenced in function "int \_\_cdecl SDL\_main(void)" (?SDL\_main@@YAHXZ)

2>Main.obj : error LNK2019: unresolved external symbol \_SDL\_Init referenced in function "int \_\_cdecl SDL\_main(void)" (?SDL\_main@@YAHXZ)

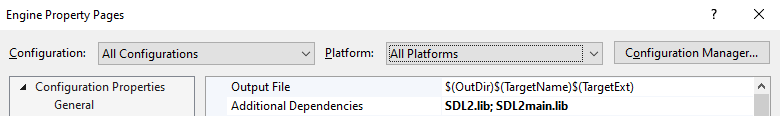
2>Main.obj : error LNK2019: unresolved external symbol \_SDL\_Quit referenced in function "int \_\_cdecl SDL\_main(void)" (?SDL\_main@@YAHXZ)

2>MSVCRTD.lib(exe\_main.obj) : error LNK2019: unresolved external symbol \_main referenced in function "int \_\_cdecl invoke\_main(void)" (?invoke\_main@@YAHXZ)

2>C:\Users\RMaple\OneDrive - Neumont College of Computer Science\Dev\GAT150\GAT150\Debug\Game.exe : fatal error LNK1120: 4 unresolved externals

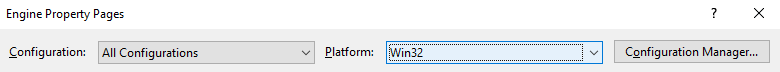
# Add SDL libraries to the Engine Project

* Open *Engine Properties* (this should be a task you are now familiar with)
* Go to the *Librarian>Additional Dependencies*
  + Add *SDL2.lib* and *SDL2main.lib*
  + Click *Apply*

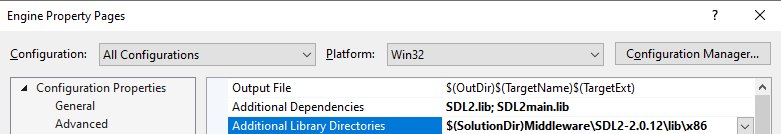


**Please follow the next instructions very carefully**

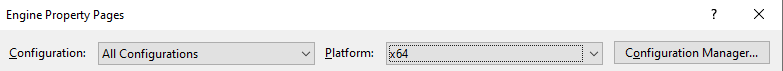
* Add the library directories but they need to be set differently based upon what platform is being built (x86/x64)
* Switch the *Platform* to *Win32*



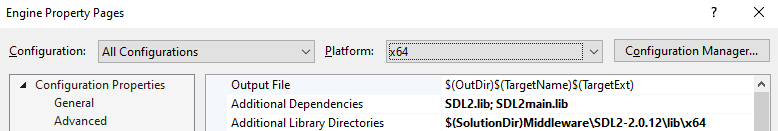
* Set the *Additional Library Directories*
  + *$(SolutionDir)Middleware\SDL2-2.0.12\lib\x86*
  + Click *Apply*

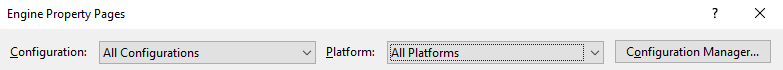


* Switch the *Platform* the *x64*



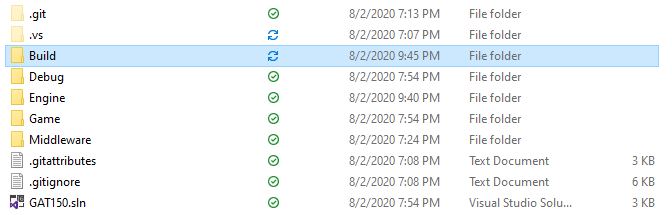
* Set the *Additional Library Directories*
  + *$(SolutionDir)Middleware\SDL2-2.0.12\lib\x64*
  + Click *Apply*



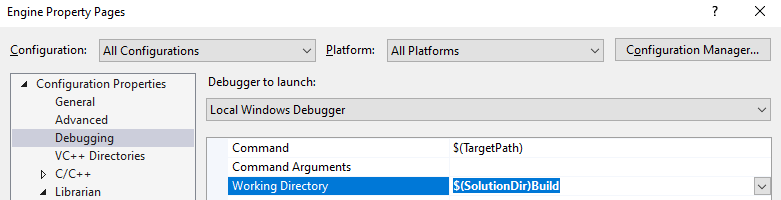
* Set the *Platform* back to *All Platforms*

# Set the Build folder

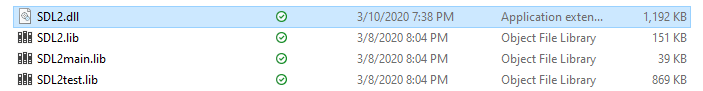
* Create a Build folder in the Solution Dir



* Set the working directory to the *Build* directory in the *Game* project
  + **This must be done to the *Game* project not the *Engine* project**
  + Select *Debugging*
  + Set the *Working Directory* to *$(SolutionDir)Build*



* Copy the SDL2.dll from the **x86** directory to the Build folder
  + *Middleware\SDL2-2.0.12\lib\x86*

******

* + **For now, we will build for the x86 platform**

****

* Build and Run the program
  + It should build and run without errors
  + A console window should appear

# Add a window

* Add the following code to create a window

if (SDL\_Init(SDL\_INIT\_VIDEO) != 0)

{

std::cout << "SDL\_Init Error: " << SDL\_GetError() << std::endl;

return 1;

}

SDL\_Window\* window = SDL\_CreateWindow("GAT150", 100, 100, 800, 600, SDL\_WINDOW\_SHOWN);

if (window == nullptr) {

std::cout << "SDL\_CreateWindow Error: " << SDL\_GetError() << std::endl;

SDL\_Quit();

return 1;

}

// wait for keyboard enter to exit

std::getchar();

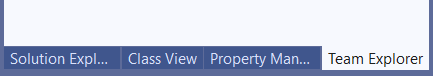
SDL\_Quit();

* Run the program and a window should be displayed

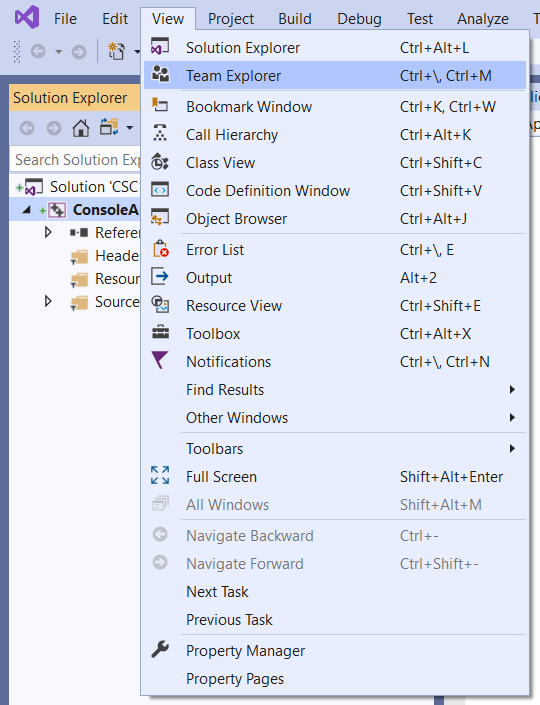


# Save the changes to GitHub

* Save the changes to the project to the remote repository (GitHub)
  + Select the *Team Explorer* tab at the bottom of the *Solution Explorer*



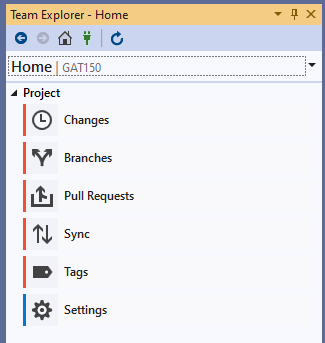
* + If the *Team Explorer* tab isn’t visible, open it with View>Team Explorer (Ctrl+\, Ctrl+M)



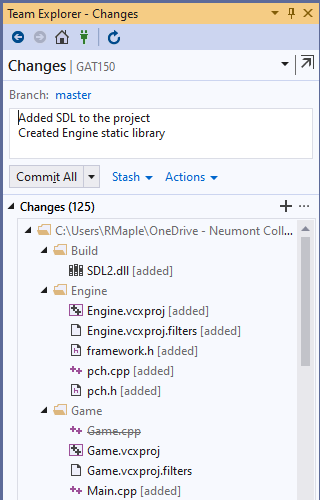
* Click the *Home* icon



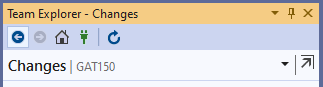
* Click the *Changes* option



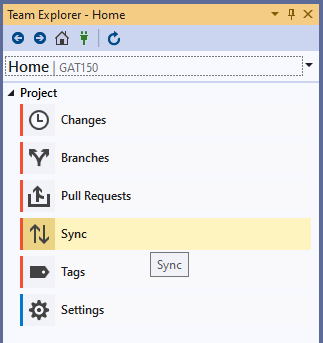
* Add a description to the change and press *Commit All*



* Press the Back or Home button



* Select the *Sync* option



* *Push* the changes in *Outgoing Commits*

