

ΔΗΜΟΚΡΙΤΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΡΑΚΗΣ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ



Εισαγωγή στα Γραφικά Υπολογιστών

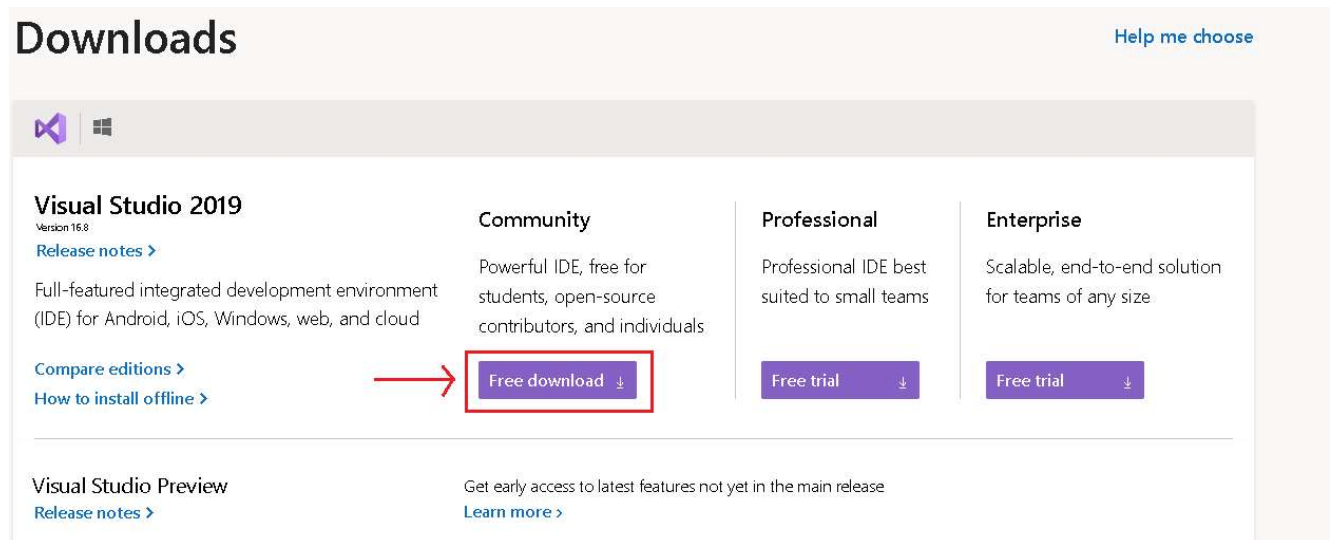
Εγκατάσταση OpenGL

Visual Studio 2019

- Αρχικά θα χρειαστούμε ένα προγραμματιστικό περιβάλλον για να δουλέψουμε.
- Αυτό είναι το Visual Studio 2019 το οποίο υποστηρίζει πολλές γλώσσες προγραμματισμού και θα σας βοηθήσει στο να έχετε μία πολύ καλή εμπειρία πάνω στο αντικείμενο του προγραμματισμού.
- Επίσης με το Visual Studio μπορούμε πολύ εύκολα να συνδέσουμε βιβλιοθήκες στο Project μας όπως θα δούμε και στην συνέχεια.

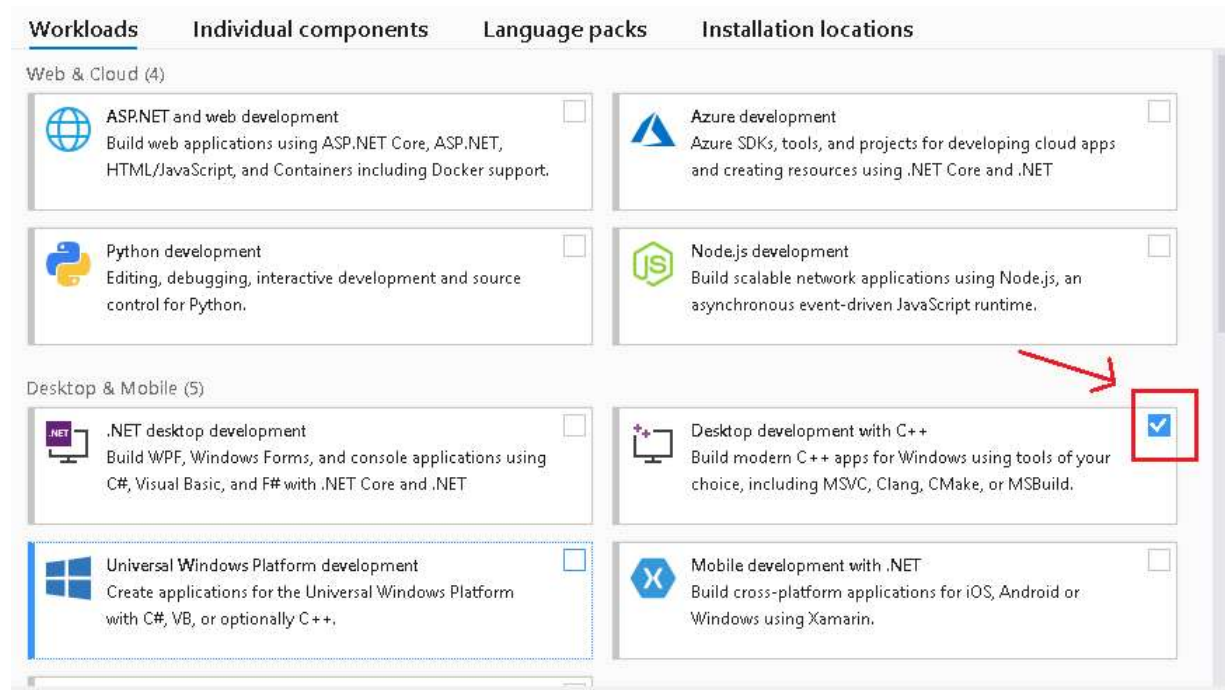
Εγκατάσταση Visual Studio 2019

- Μεταβείτε στον σύνδεσμο : <https://visualstudio.microsoft.com/downloads/> για να κατεβάσετε το Visual Studio 2019.



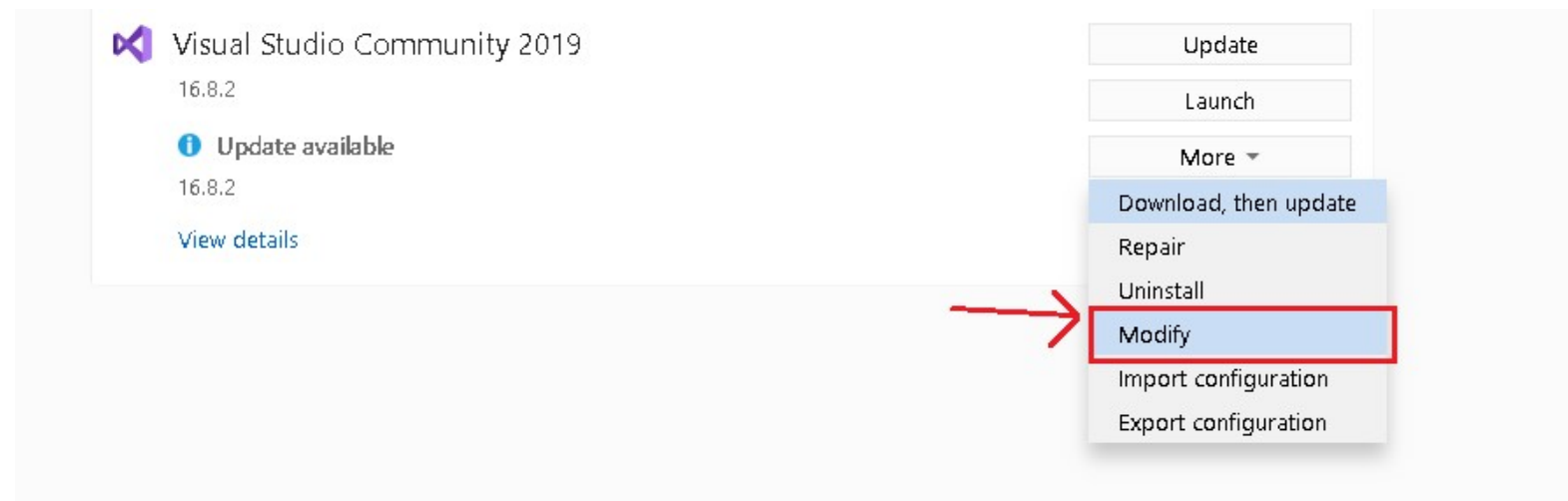
Εγκατάσταση Visual Studio 2019

- Ανοίξτε το .exe αρχείο που μόλις κατεβάσατε και όταν εμφανιστεί το παρακάτω παράθυρο επιλέξτε Desktop development with C++ εφόσον θα προγραμματίσουμε σε C++ και στη συνέχεια install.



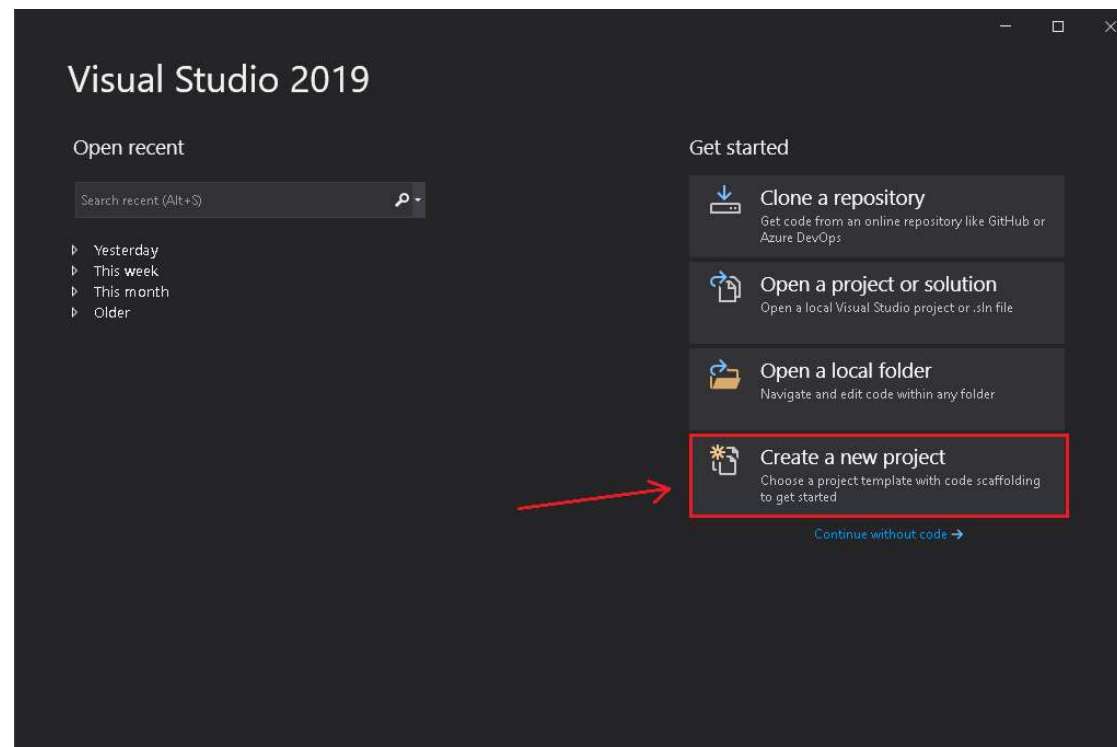
Εγκατάσταση Visual Studio 2019

- Στην περίπτωση που θέλετε να εγκαταστήσετε και άλλα διαθέσιμα πακέτα (π.χ. Game development with Unity) μην διαγράψετε το .exe αρχείο (για να μην χρειαστεί να το ξανά κατεβάστε) και ανοίξτε το, αλλά αυτήν την φορά εφόσον το Visual Studio είναι ήδη εγκατεστημένο θα χρειαστεί να επιλέξετε modify και θα εμφανιστεί το παράθυρο όπως στην προηγούμενη διαφάνεια.



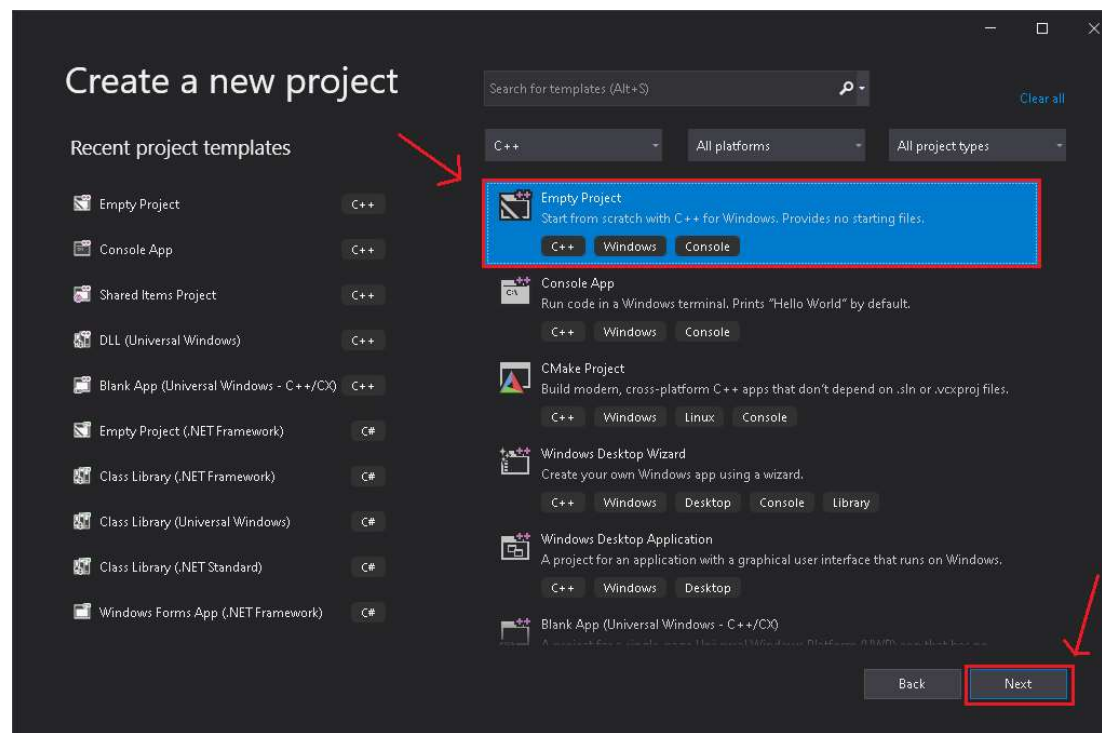
Δημιουργία Project Visual Studio 2019

- Εφόσον έχετε εγκαταστήσει το Visual Studio ανοίξτε το πρόγραμμα και θα εμφανιστεί το παρακάτω παράθυρο. Επίλέξτε Create New Project.



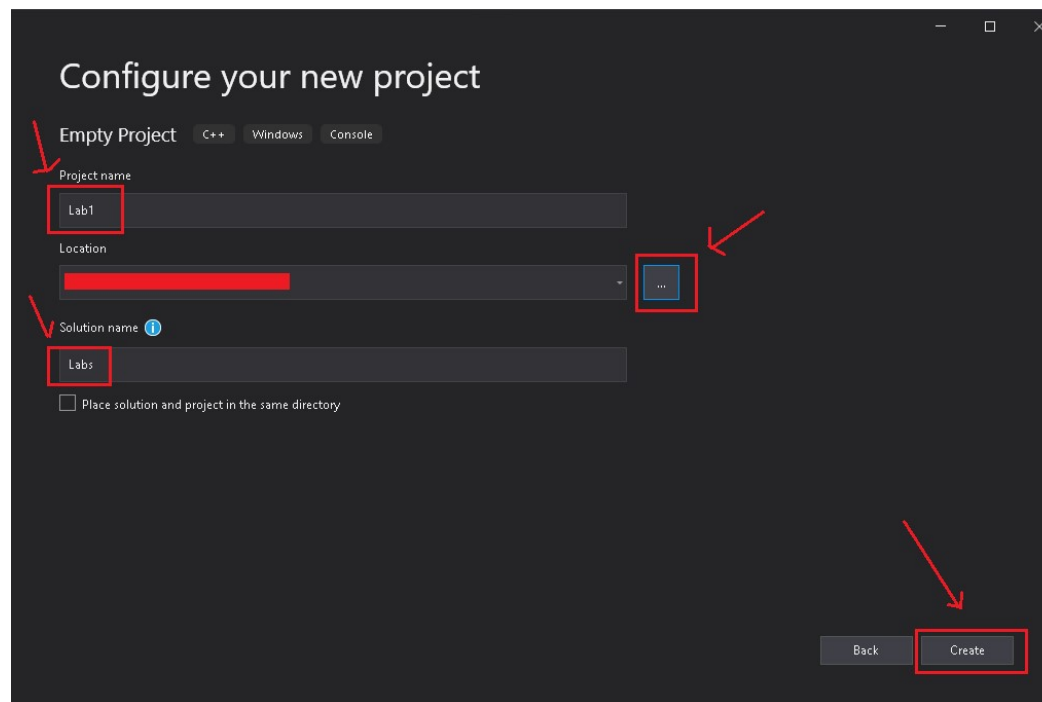
Δημιουργία Project Visual Studio 2019

- Στην συνέχεια επιλέγουμε empty C++ Project και next. Αν δεν είχαμε επιλέξει πιο πριν 'Desktop development with C++' δεν θα υπήρχε αυτή η επιλογή!!!



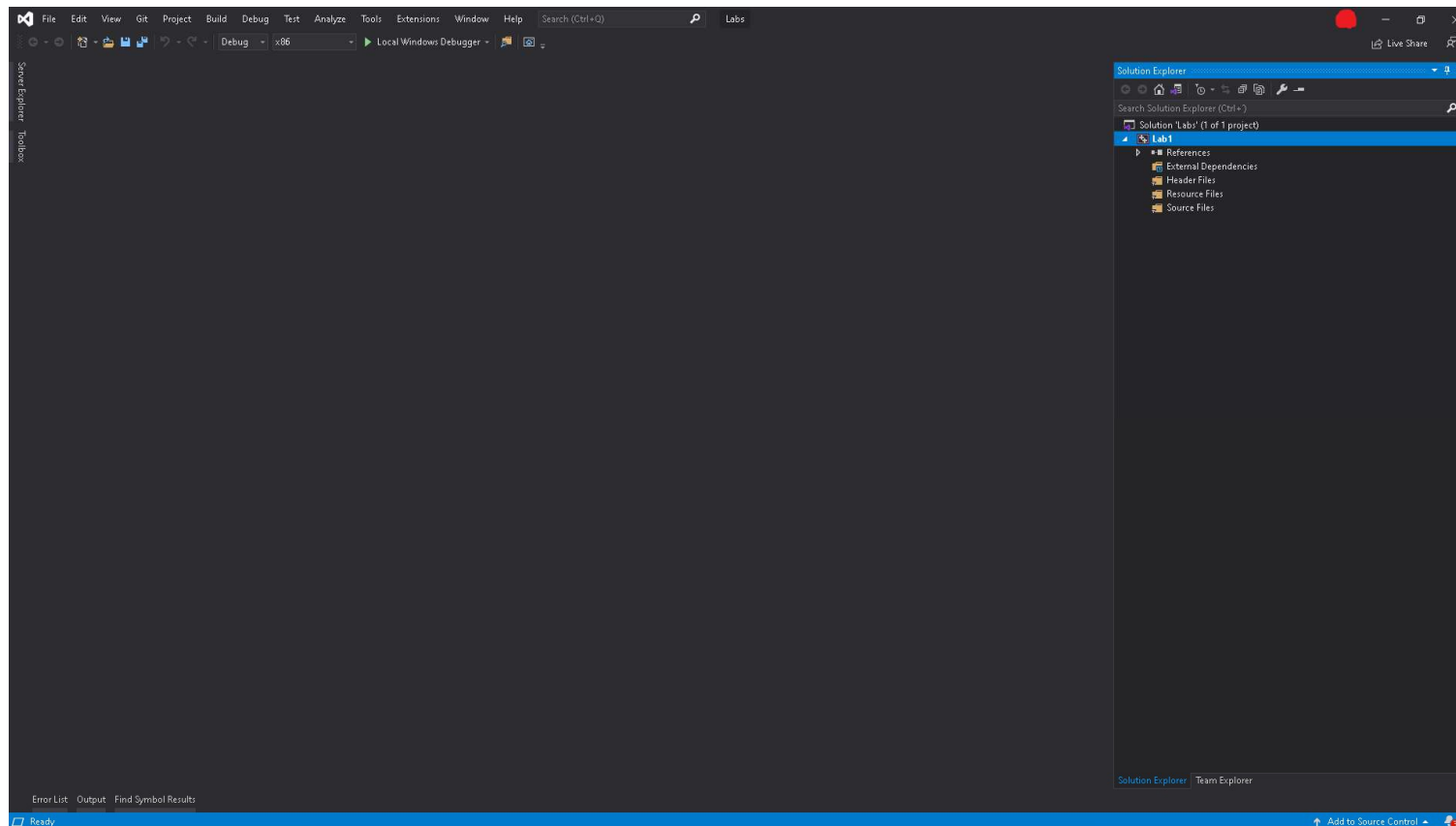
Δημιουργία Project Visual Studio 2019

- Μετονομάζουμε το Project name σε 'Lab1', επιλέγουμε τον φάκελο που θέλουμε να αποθηκεύσουμε το Project, μετονομάζουμε το Solution name σε 'Labs' και τέλος επιλέγουμε Create.



Διαχείριση Project Visual Studio 2019

- Όταν Δημιουργηθεί το Project θα δείτε το περιβάλλον όπως φαίνεται στην παρακάτω εικόνα.

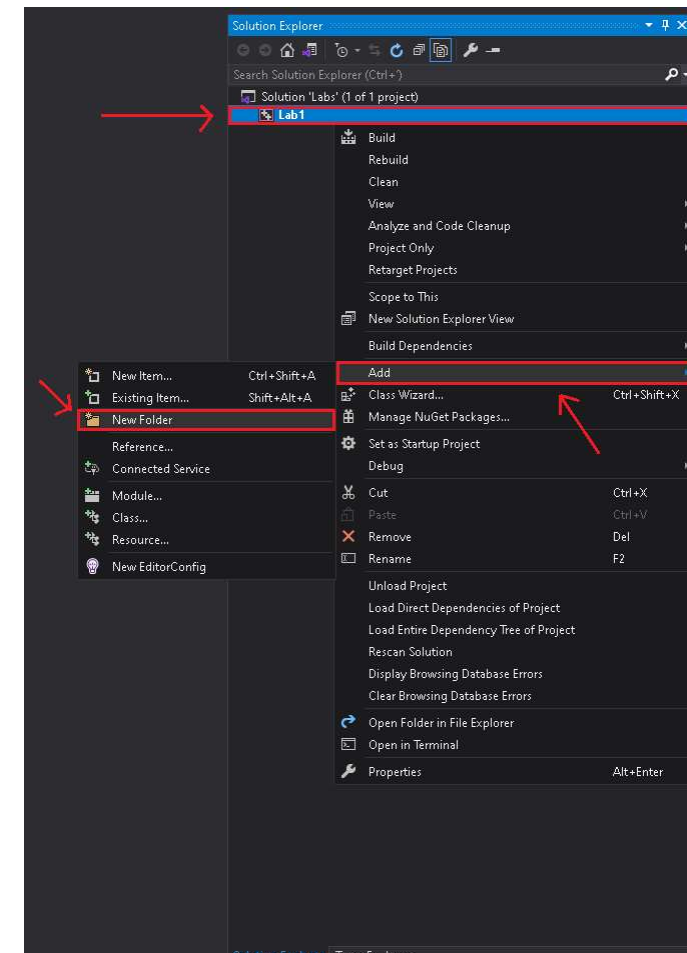


Διαχείριση Project Visual Studio 2019

- Παρατηρούμε ότι στον Solution Explorer (πάνω δεξιά) υπάρχει το Solution με το όνομα που δώσαμε πιο πριν 'Labs' και μέσα στο Solution υπάρχει το Project με το όνομα που δώσαμε πιο πριν (Lab1).
- Επίσης μέσα στον φάκελο του Project 'Lab1' εντοπίζουμε 4 φακέλους : 'External Dependencies', 'Header Files', 'Resource Files' και 'Source Files'. Αυτά δεν είναι φάκελοι αλλά εικονικά φίλτρα τα οποία είναι διαθέσιμα μόνο όταν είναι ανοιχτό το Project και δεν θα τα βρείτε σε φακέλους του υπολογιστή.
- Έτσι θα τροποποιήσουμε λίγο τον Solution Explorer έτσι ώστε να δημιουργήσουμε φακέλους που θα είναι ορατοί, όπως θα δείτε στην επόμενη διαφάνεια.

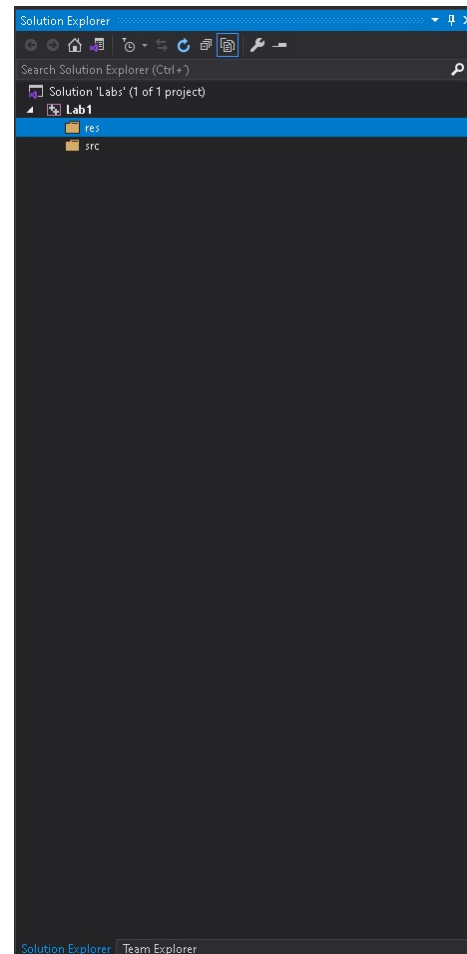
Διαχείριση Project Visual Studio 2019

- Αρχικά επιλέγουμε 'Show All Files' και παρατηρούμε ότι εξαφανίζονται τα φίλτρα και εμφανίζετε και ένα κουμπί refresh.
- Στην συνέχεια δεξί κλικ στο project 'Lab1', επιλογή 'Add' και επιλογή 'New Folder'



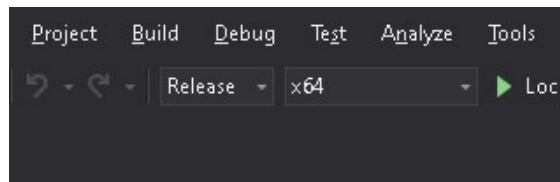
Διαχείριση Project Visual Studio 2019

- Δημιουργούμε δύο φακέλους με τα ονόματα 'src' και 'res'.
- Στον φάκελο 'src' (σημαίνει source code) θα τοποθετούμε τα αρχεία πηγαίου κώδικα δηλαδή τα .cpp και .h αρχεία.
- Στον φάκελο 'res' (σημαίνει resources) θα τοποθετούμε αρχεία όπως shaders, εικόνες, μοντέλα κ.τ.λ.π.



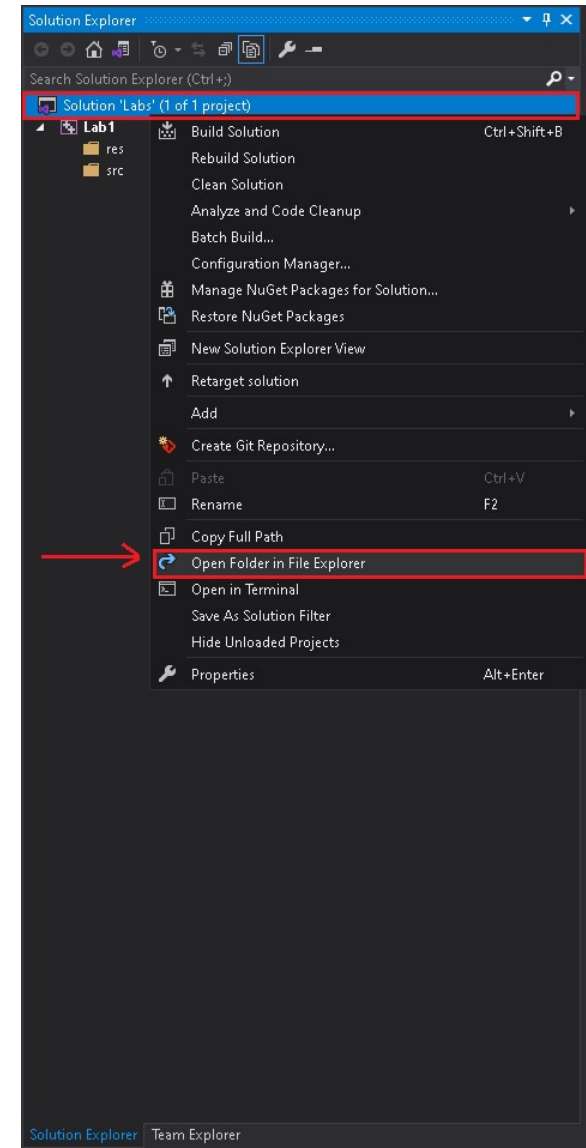
Διαχείριση Project Visual Studio 2019

- Στη συνέχεια αλλάζουμε την πλατφόρμα του Solution από x86 σε x64 και το Configuration από Debug σε Release.
- Το x86 σημαίνει ότι η εφαρμογή μας θα τρέχει σε 32 bits ενώ το x64 θα τρέχει σε 64 bits. Η επιλογή αυτή έχει να κάνει με το ότι οι βιβλιοθήκες που θα συνδέσουμε στην συνέχεια επιλέχθηκαν για 64 bits. Θα μπορούσαμε να κατεβάσουμε τις βιβλιοθήκες για 32 bits και επιλέξουμε x86. Γενικά σε ένα 64 bits σύστημα μπορούμε να τρέξουμε εφαρμογές σε 64 αλλά και σε 32 bits ενώ σε ένα 32 bits σύστημα μπορούμε να τρέξουμε μόνο 32 bits εφαρμογές.
- Η διαφορά του Debug mode με Release mode έχει να κάνει με το ότι στο Release mode πραγματοποιείται βελτιστοποίηση της εφαρμογής και πολύ πιθανόν να παραληφθούν μέρη του κώδικα που δεν είναι χρήσιμα με αποτέλεσμα τρέχει πιο γρήγορα η εφαρμογή. Στο Debug mode μπορούμε πιο εύκολα να εντοπίσουμε λάθη όταν κάνουμε Debugging γιατί δεν πραγματοποιείται κάποια βελτιστοποίηση.



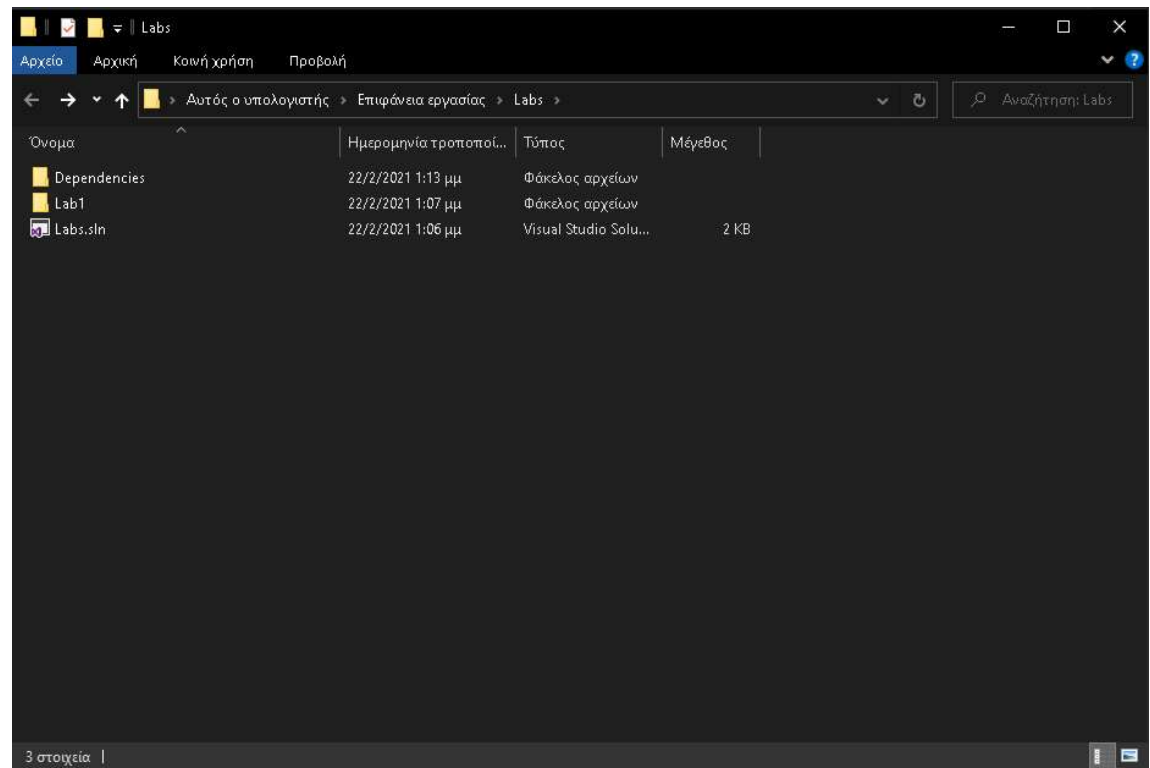
Τοποθέτηση αρχείων

- Για την τοποθέτηση των βιβλιοθηκών κατεβάζουμε το αρχείο OpenGLInstallation.zip από το eclass.
- Το Κάνουμε unzip και θα δούμε έναν φάκελο Dependencies, ένα αρχείο .cpp και κάποια .txt αρχεία.
- Πηγαίνουμε στο Visual Studio και με δεξί κλικ στο Solution 'Labs' επιλέγουμε 'Open Folder In File Explorer'.



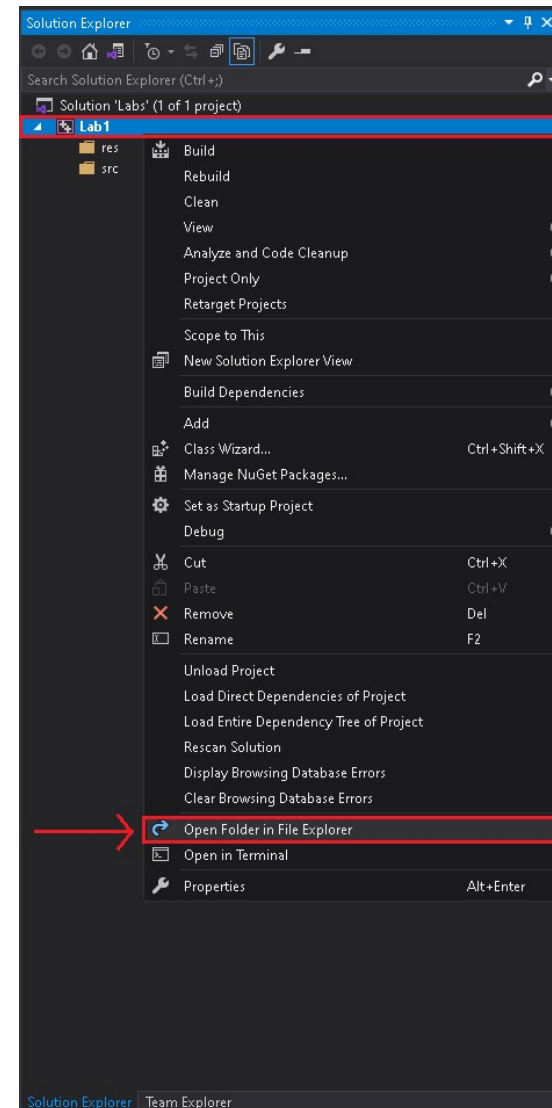
Τοποθέτηση αρχείων

- Αυτόματα ανοίγει ο φάκελος του Solution και παρατηρούμε ότι υπάρχει ένα .sln αρχείο και ο φάκελος 'Lab1' που είναι ο φάκελος του Project μας. Στον φάκελο του Solution τοποθετούμε τον φάκελο Dependencies.
- Γενικά για να ανοίξουμε το Visual Studio και να επεξεργαστούμε το Project μας κάνουμε διπλό κλικ στο .sln αρχείο.



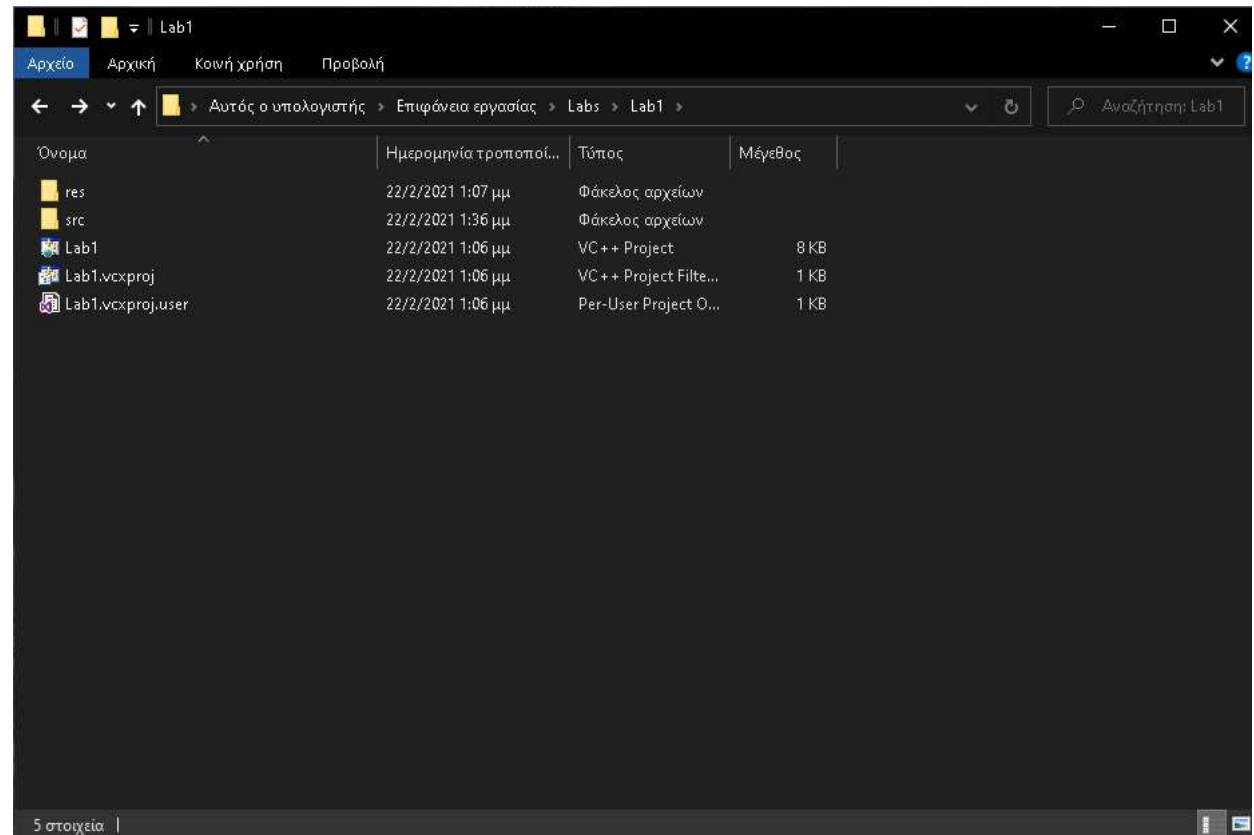
Τοποθέτηση αρχείων

- Στην συνέχεια πηγαίνουμε στο Visual Studio και με δεξί κλικ στο Project 'Lab1' επιλέγουμε 'Open Folder In File Explorer'.



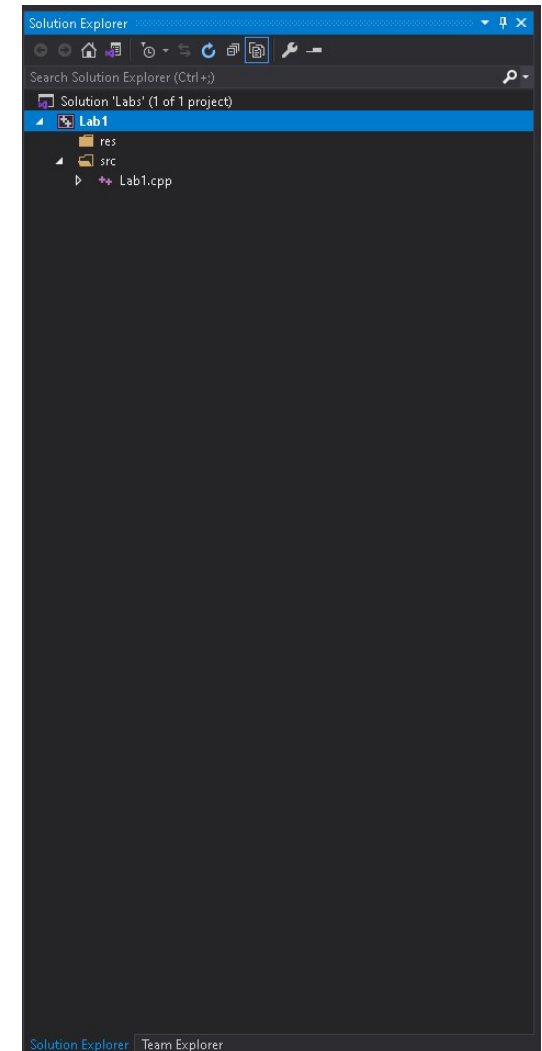
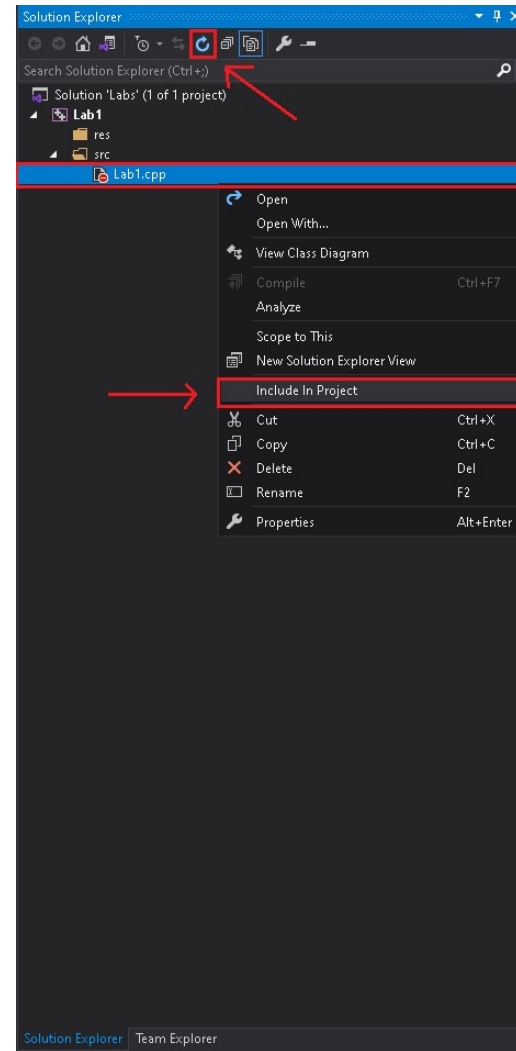
Τοποθέτηση αρχείων

- Παρατηρούμε ότι ο φάκελος αυτός περιέχει τους φακέλους 'res' και 'src' που δημιουργήσαμε πριν. Επίσης είναι ο ίδιος φάκελος 'Lab1' που είχαμε δει πριν στον φάκελο του Solution.
- Τοποθετούμε το αρχείο 'Lab1.cpp' μέσα στον φάκελο 'src'.



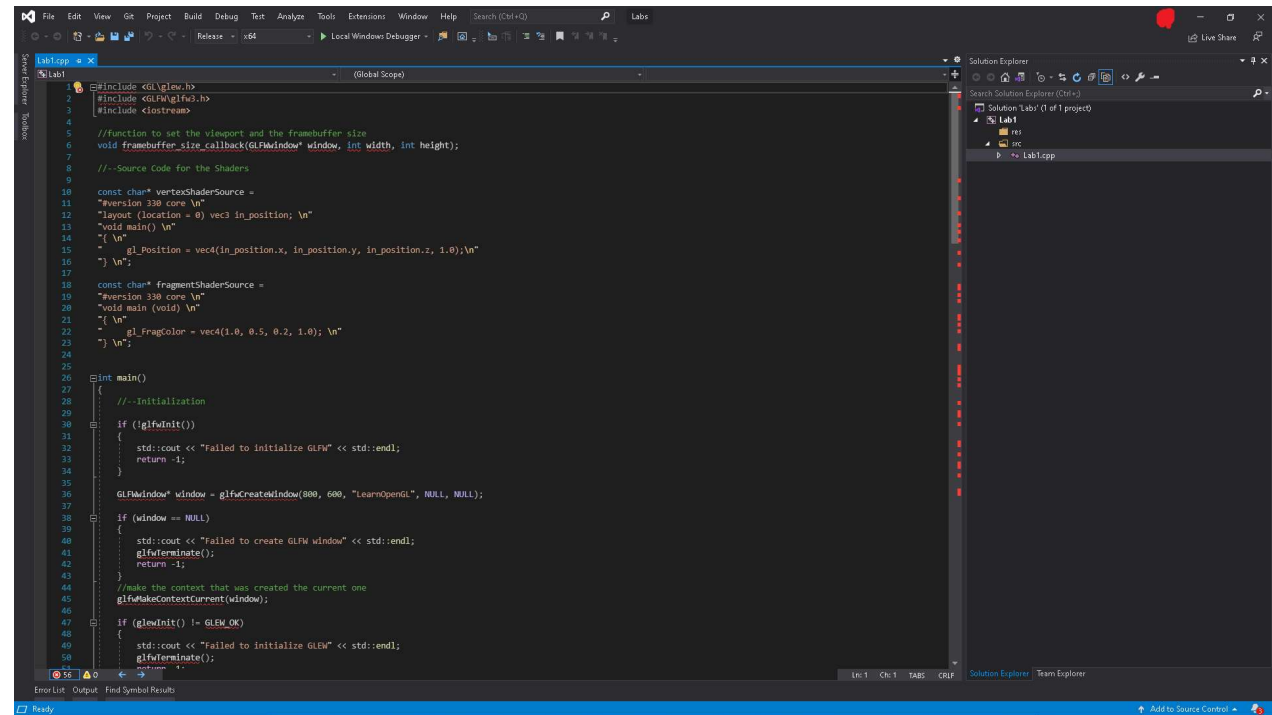
Τοποθέτηση αρχείων

- Στη συνέχεια πηγαίνουμε πίσω στο Visual Studio και κάνουμε refresh. Παρατηρούμε ότι εμφανίζεται το αρχείο 'Lab1.cpp' μέσα στον φάκελο 'src'.
- Επίσης με δεξί κλικ στο 'Lab1.cpp' επιλέγουμε 'Include In Project'.
- Παρατηρούμε ότι αλλάζει το εικονίδιο στο αρχείο 'Lab1.cpp'. (το φίλτρο 'External Dependencies' αγνοείστε το αν εμφανιστεί).



Σύνδεση Βιβλιοθηκών

- Ανοίγουμε το αρχείο 'Lab1.cpp' με διπλό κλικ.
- Παρατηρούμε ότι πολλές γραμμές κώδικα έχουν κόκκινη υπογράμμιση και αυτό σημαίνει επειδή δεν έχουμε συνδέσει τις βιβλιοθήκες από τον φάκελο 'Dependencies' που είδαμε πριν.



```
#include <gl.h>
#include <glfw3.h>
#include <iostream>

//function to set the viewport and the framebuffer size
void framebuffer_size_callback(GLFWwindow* window, int width, int height);

//Source Code for the Shaders
const char* vertexShaderSource =
"version 330 core\n"
"layout (location = 0) vec3 in_position; \n"
void main() \n"
{ \n"
gl_Position = vec4(in_position.x, in_position.y, in_position.z, 1.0);\n"
} \n";

const char* fragmentShaderSource =
"version 330 core\n"
void main (void) \n"
{ \n"
gl_FragColor = vec4(1.0, 0.5, 0.2, 1.0);\n"
} \n";

int main()
{
//--Initialization
if (!glfwInit())
{
std::cout << "Failed to initialize GLFW" << std::endl;
return -1;
}

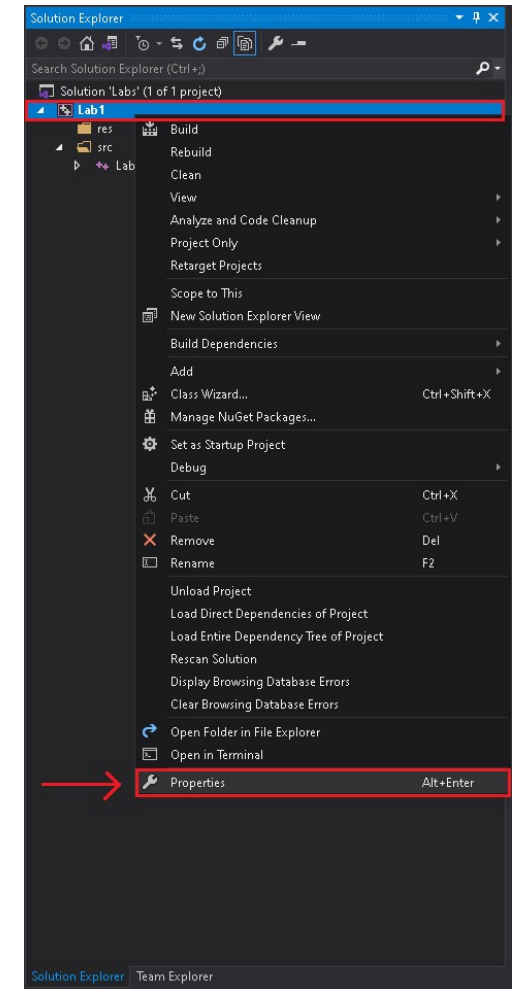
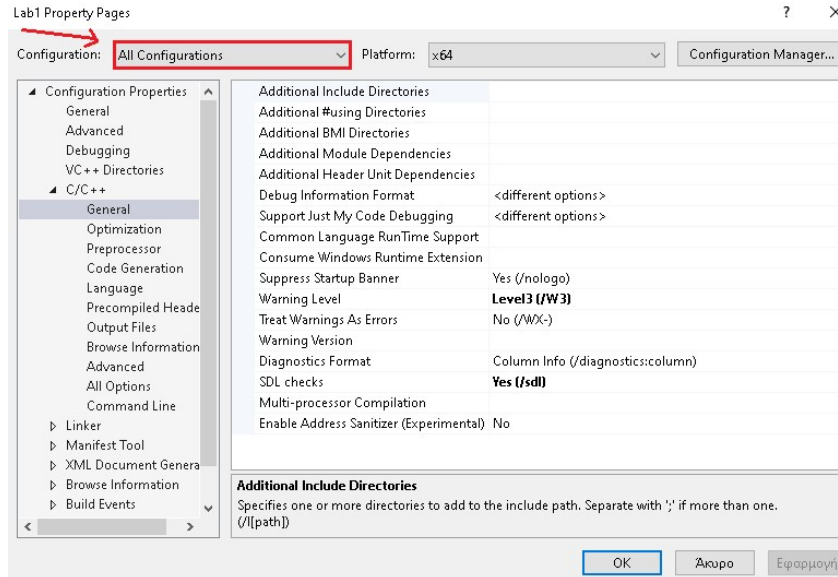
GLFWwindow* window = glfwCreateWindow(800, 600, "LearnOpenGL", NULL, NULL);
if (window == NULL)
{
std::cout << "Failed to create GLFW window" << std::endl;
glfwTerminate();
return -1;
}

//make the context that was created the current one
glfwMakeContextCurrent(window);

if (glewInit() != GLEW_OK)
{
std::cout << "Failed to initialize GLEW" << std::endl;
glfwTerminate();
}
```

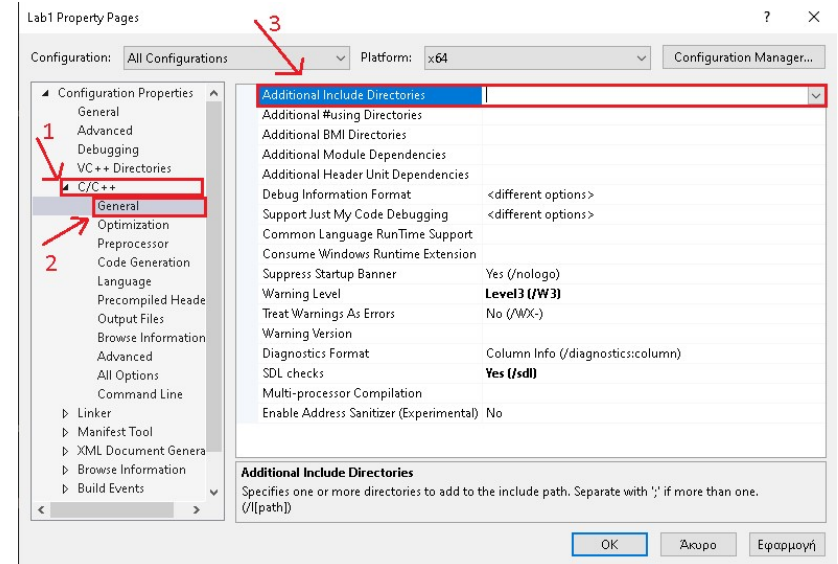
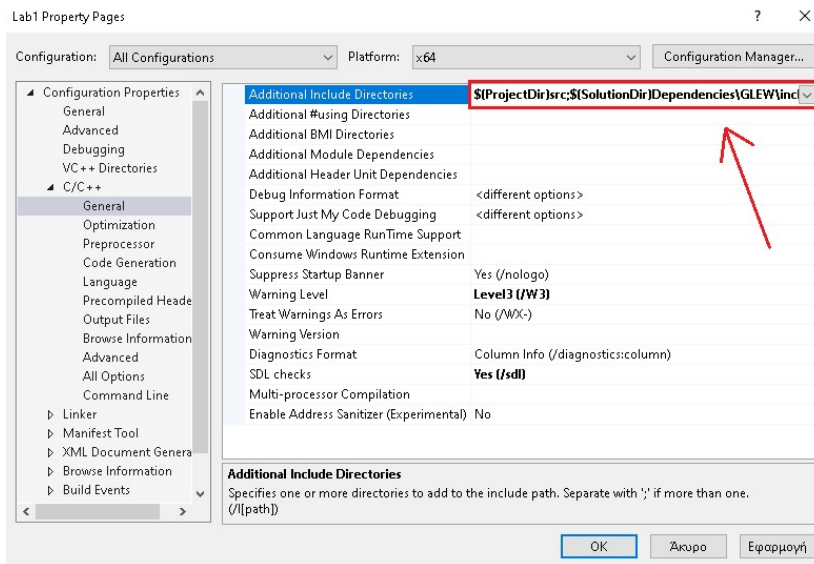
Σύνδεση Βιβλιοθηκών

- Με δεξί κλικ στο Project επιλέγουμε 'Properties'.
- Στη συνέχεια 'All Configurations'. Είναι σημαντικό διότι μπορεί να εναλλάσσουμε το Debug με το Release mode και θέλουμε οι αλλαγές να αποθηκευτούν και για τα δύο Configurations.
- Για τα Platforms δεν χρειάζεται εφόσον οι βιβλιοθήκες μας είναι για 64 bits.



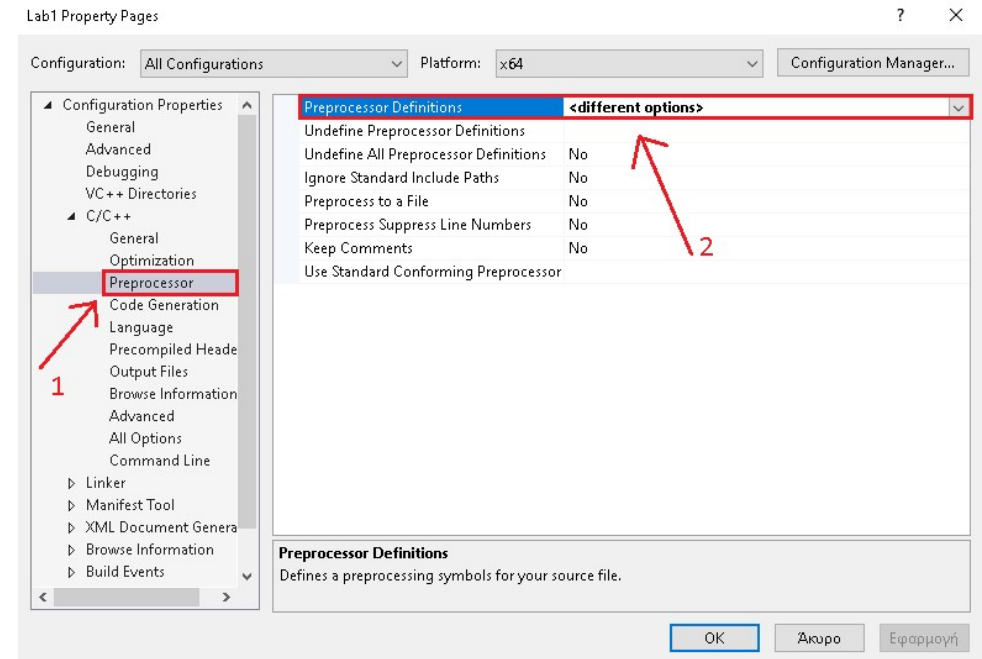
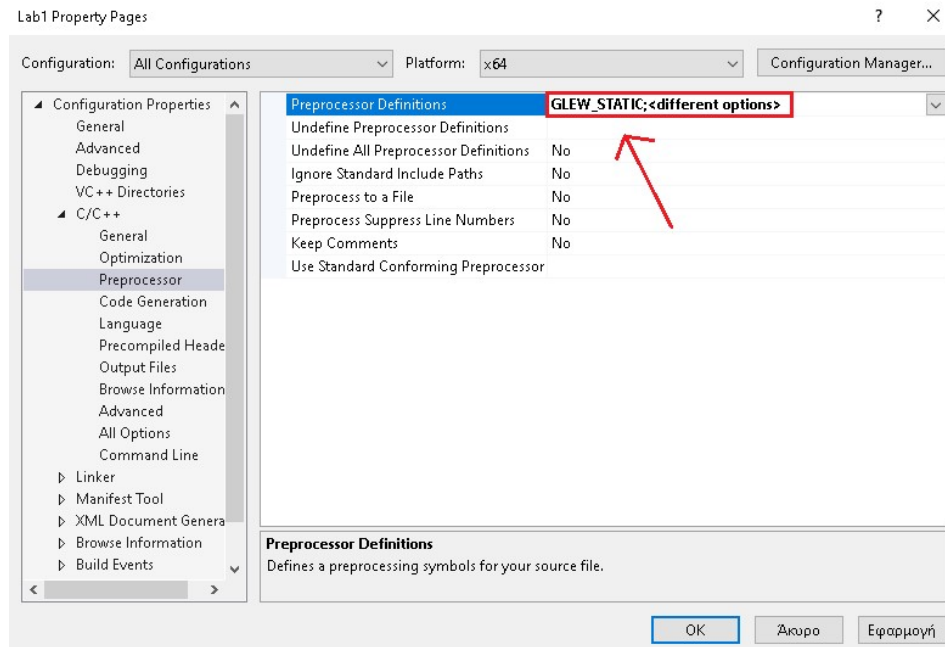
Σύνδεση Βιβλιοθηκών

- Στη συνέχεια μεταβαίνουμε στο C/C++ -> General -> Additional Include Directories.
- Κάνουμε copy paste ότι υπάρχει μέσα στο 'Additional Include Directories.txt' όπως φαίνεται και στην εικόνα.



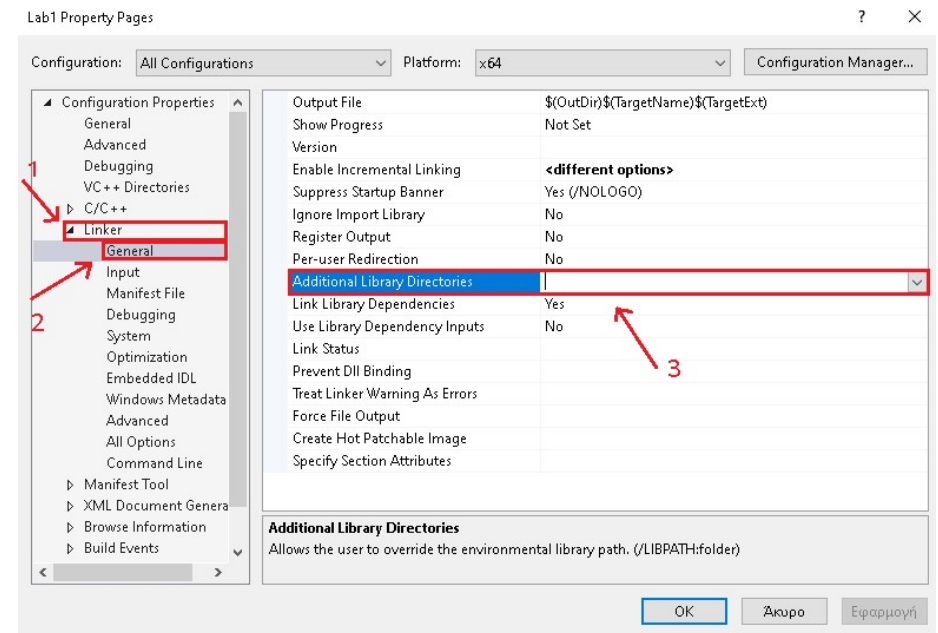
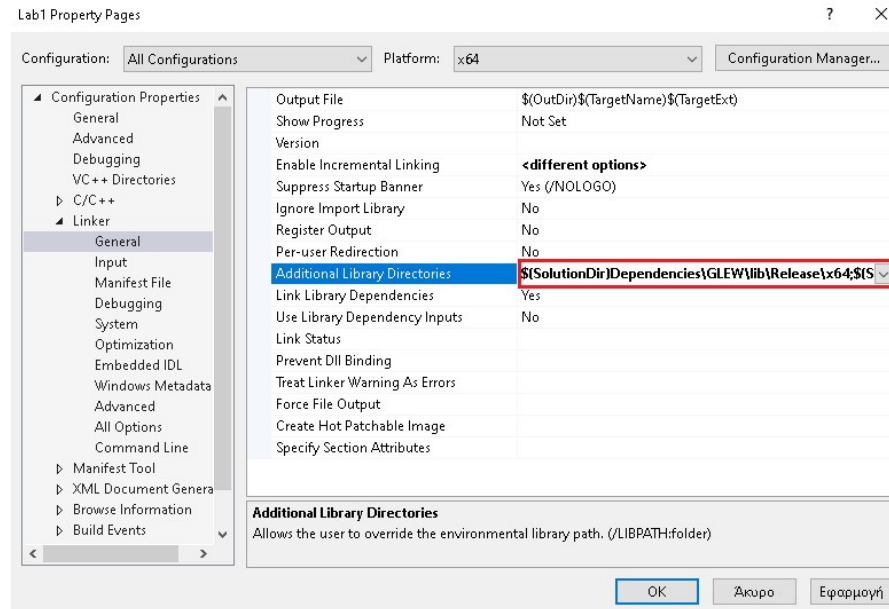
Σύνδεση Βιβλιοθηκών

- Εφόσον έχουμε επιλέξει το 'C/C++' μεταβαίνουμε στο Preprocessor -> Preprocessor Definitions.
- Γράφουμε GLEW_STATIC; όπως φαίνεται στην εικόνα.

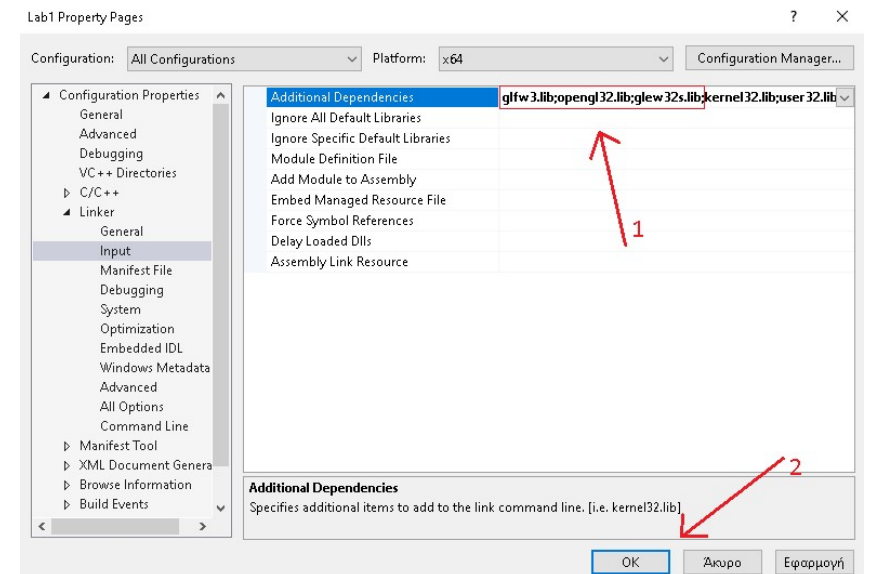


Σύνδεση Βιβλιοθηκών

- Μεταβαίνουμε στο Linker -> General -> Additional Library Directories.
- Κάνουμε copy paste ότι υπάρχει μέσα στο 'Additional Library Directories.txt' όπως φαίνεται και στην εικόνα.



- Εφόσον έχουμε επιλέξει το 'Linker' μεταβαίνουμε στο Input -> Additional Dependencies
- Κάνουμε copy paste ότι υπάρχει μέσα στο 'Additional Library Directories.txt' στην αρχή μαζί με όλα τα άλλα αρχεία όπως φαίνεται και στην εικόνα.
- Τέλος πατάμε 'ΟΚ'.



Τελικό Αποτέλεσμα

- Αν έχετε ακολουθήσει όλα τα βήματα πατώντας F5 η εφαρμογή πρέπει να τρέξει και να πάρετε το παρακάτω αποτέλεσμα.

