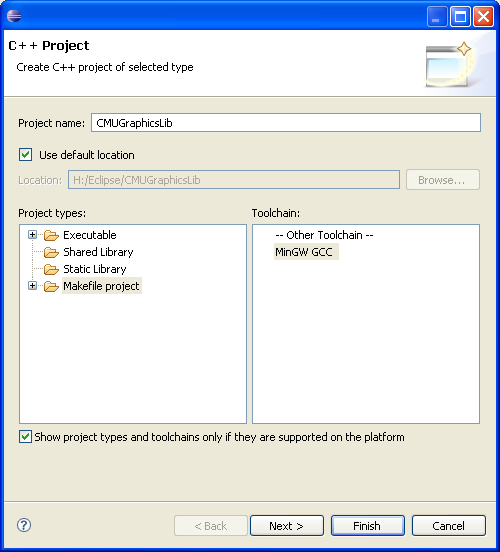
# **Using the Carnegie Mellon Graphics Library**

This guide details how to use the Carnegie Mellon Graphics Library (CMUGraphicsLib) in Eclipse. This guide details the instructions for creating a CMUGraphicsLib project, as well as importing the sample project and configuring it use the library

## ***Importing CMUGraphicsLib***

1. Launch Eclipse and switch to C++ mode, if necessary
2. Make a new project called “CMUGraphicsLib” using the “Makefile project” defaults, selecting the “MinGW GCC” Toolchain



1. Extract the contents of “CMUGraphicsLib-Eclipse.zip” to the CMUGraphicsLib directory in your workspace. Press File 🡪 Refresh
2. Go to Window 🡪 Preferences
   1. Expand C/C++, then New CDT Project Wizard, then Makefile Project
   2. Go to the Builder settings tab
   3. Uncheck use default build command and enter “mingw32-make.exe” into the Build command text box. Press OK
3. Right-click on CMUGraphicsLib, press properties, C/C++ Build, and repeat the above steps.
4. Press “Build”
5. CMUGraphicsLib is now ready for use

## ***Creating a CMUGraphicsLib Project***

Now that the CMUGraphicsLib library is built, it now can be used by other code to do graphics. However, a special project must first be created

1. File 🡪 New 🡪 C++ Project and name it. For sample purposes, call it “CMU Demo”
2. Right-click on it and press Properties. Go to the C/C++ Build 🡪 Settings
   1. Under GCC C++ Compiler, select Directories, Press add, press workspace, select CMUGraphicsLib, and choose the CMUGraphicsLib folder in the project. Press OK twice
   2. Under MinGW C++ Linker, do the following:
      1. In the top box, press add, and type “cmugraphics” and press OK
      2. Also in the top box, press add and type “gdi32” and press OK
      3. In the bottom box, press add, press workspace, select CMUGraphicsLib and press OK twice.
   3. Press OK to close the dialog.
3. Now, you can write code. For an example program, paste the following in house.cpp and run it:

**#include** <iostream>

**#include** <cmath>

**#include** <CMUgraphics.h>

**using** **namespace** std;

// global constants

**const** **int** HOUSE\_WIDTH = 400;

**const** **int** HOUSE\_HEIGHT = 200;

**const** **int** DOOR\_WIDTH = 30;

**const** **int** DOOR\_HEIGHT = 50;

**const** **int** WINDOW\_WIDTH = 50;

**const** **int** WINDOW\_HEIGHT = 70;

// declarations

**void** **DrawHouse**();

**void** **DrawFrame**(**int** x, **int** y);

**void** **DrawDoor**(**int** x, **int** y);

**void** **DrawWindows**();

**void** **DrawWindow**();

// global window object, 700x400 at (5,5)

window myWindow(700, 400, 5, 5);

**int** **main**() {

cout << "Hit <enter> once you've resized and moved the Console";

cout << " window out of the way...\n";

cin.ignore();

DrawHouse();

**return** 0;

}

**void** **DrawHouse**() {

**int** x, y;

cout << "You will be asked for the coordinates of the top left corner of the house.\n";

cout << "100,150 is a good choice.\n\n";

cout << "Enter x coordinate of house (top left): ";

cin >> x;

cout << "Enter y coordinate of house (top left): ";

cin >> y;

DrawFrame(x,y);

DrawDoor(x,y);

DrawWindows();

}

**void** **DrawFrame**(**int** x, **int** y) {

myWindow.SetPen(BROWN);

myWindow.SetBrush(BROWN);

myWindow.DrawRectangle(x, y, x + HOUSE\_WIDTH, y + HOUSE\_HEIGHT, *FILLED*);

myWindow.SetPen(BLACK);

myWindow.DrawLine(x, y, x + HOUSE\_WIDTH/2, y - HOUSE\_HEIGHT/2);

myWindow.DrawLine(x + HOUSE\_WIDTH, y, x + HOUSE\_WIDTH/2, y - HOUSE\_HEIGHT/2);

}

**void** **DrawDoor**(**int** x, **int** y) {

myWindow.SetPen(BLACK);

myWindow.SetBrush(BLACK);

myWindow.DrawRectangle(x + HOUSE\_WIDTH/2 - DOOR\_WIDTH/2, y + HOUSE\_HEIGHT - DOOR\_HEIGHT,

x + HOUSE\_WIDTH/2 + DOOR\_WIDTH/2, y + HOUSE\_HEIGHT, *FILLED*);

}

**void** **DrawWindows**() {

**int** i, n;

cout << "How many windows? ";

cin >> n;

**for** (i = 0; i < n; i++) {

DrawWindow();

}

}

**void** **DrawWindow**() {

**int** x, y;

cout << "Click top left corner of window location." << **endl**;

myWindow.WaitMouseClick(x, y);

myWindow.SetPen(BLACK);

myWindow.SetBrush(WHITE);

myWindow.DrawRectangle(x, y, x + WINDOW\_WIDTH, y + WINDOW\_HEIGHT, *FILLED*);

myWindow.SetPen(BLACK);

myWindow.DrawLine(x, y + WINDOW\_HEIGHT/2, x + WINDOW\_WIDTH, y + WINDOW\_HEIGHT / 2);

myWindow.DrawLine(x + WINDOW\_WIDTH/2, y, x + WINDOW\_WIDTH/2, y + WINDOW\_HEIGHT);

}