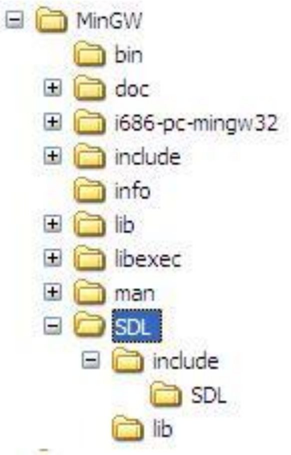
Introduction to SDL

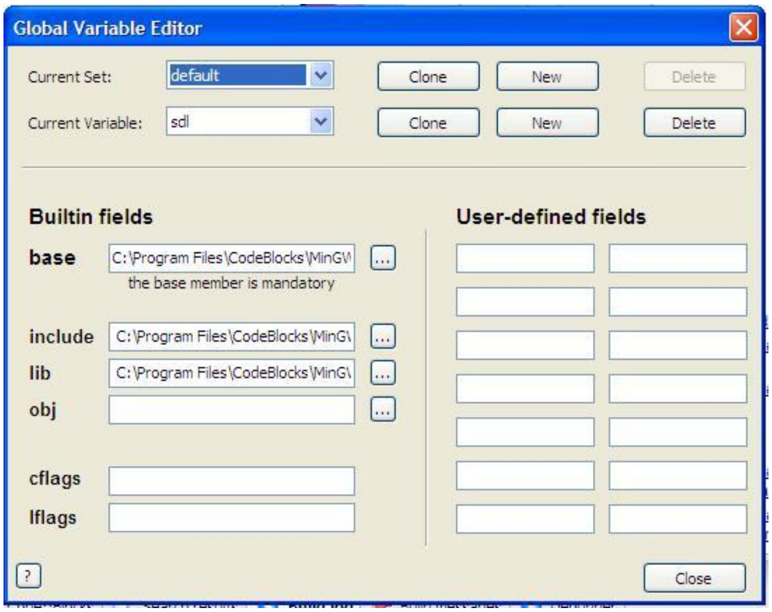
1) Background
    a) Initially created by Sam Lantinga at Loki Intertainment
        i) He is still maintaining it, but primary job is with Blizzard
    b) Designed as cross platform graphical C library
        i) What is graphical library
            (1) Lowest layer is hardware
            (2) Next is OS
            (3) Next is graphics library
            (4) Finally application
        ii) Why use graphical libraries
            (1) Portability
                (a) Don't have to worry about hardware specifics
                (b) Supported on multiple systems
            (2) Ease of development
                (a) Simplifies coding so you can focus on what you want, not on how to talk to video or audio cards
        iii) Alternatives
            (1) DirectX
                (a) Microsoft for Windows and Xbox
                (b) Current version is 11
                (c) Developed after fiasco with people writing directly to hardware in early windows systems
                (d) http://www.gamesforwindows.com/en-US/AboutGFW/Pages/DirectX10.aspx
            (2) OpenGL
                (a) Initially developed at Silicon Graphics (SGI)
                (b) Now maintained by OpenGL.org as part of Khronos group
                (c) Cross platform standard
                (d) http://www.opengl.org/
        iv) Why SDL at LCC
            (1) Not Microsoft, provides cross platform portability
            (2) Initializing OpenGL is platform specific
            (3) OpenGL takes more lines of code to do same project
                http://www.libsdl.org/cgi/docwiki.cgi/Using_OpenGL_With_SDL

        v) What does SDL provide
            (1) Initializing code (and clean-up code)
            (2) Eight subsystems
                (a) Audio
                (b) CDROM
                (c) Event Handling
                (d) File I/O
                (e) Joystick handling
                (f) Threading
                (g) Timers
                (h) Video

2) Using it with Code::Blocks
   a) Install libraries on your computer
      i) Download LCC_SDL_Libs.zip
      ii) Unzip and rename the folder that was inside the zip file to “SDL”.
      iii) Move to MinGW\
         Note: if your MinGW folder is in the Codeblocks folder, then the whole path will be:
            C:\Program Files\CodeBlocks\MinGW\SDL
         Or, if your MinGW is in the root, then the whole path will be:
            C:\MinGW\SDL



   b) Set up Code::Blocks
      i) Set Global Variables
         (1) Click on “Settings”, “Global Variables”. This will open the Global Variable Editor.
         (2) In the drop down list for “Current Variable” select “sdl”
            Note: If there is no sdl variable in your variable list, then follow the instructions in 2, c, for creating an SDL project. At one of the steps in the SDL new project wizard it will open the Global Variable editor and you will be able to continue with step 3 below.
         (3) In the text box for “base”, enter the full path to your SDL folder. (The same one from step 2, a, iii).
         (4) In the text box for “include”, enter the path to the SDL include folder. If your base path was the same as the one in the example in step 2, a, iii, then you would enter:
            C:\Program Files\CodeBlocks\MinGW\SDL\include
         (5) In the text box for “lib”, enter the path to the SDL library folder. If your base path was the same as the one in the example in step 2, a, iii, then you would enter:
            C:\Program Files\CodeBlocks\MinGW\SDL\lib
         (6) Leave the rest of the entries empty or default and click “Close”.



Current Set: default

Current Variable: sdl

Builtin fields

base: C:\Program Files\CodeBlocks\MinGV

the base member is mandatory

include: C:\Program Files\CodeBlocks\MinG\

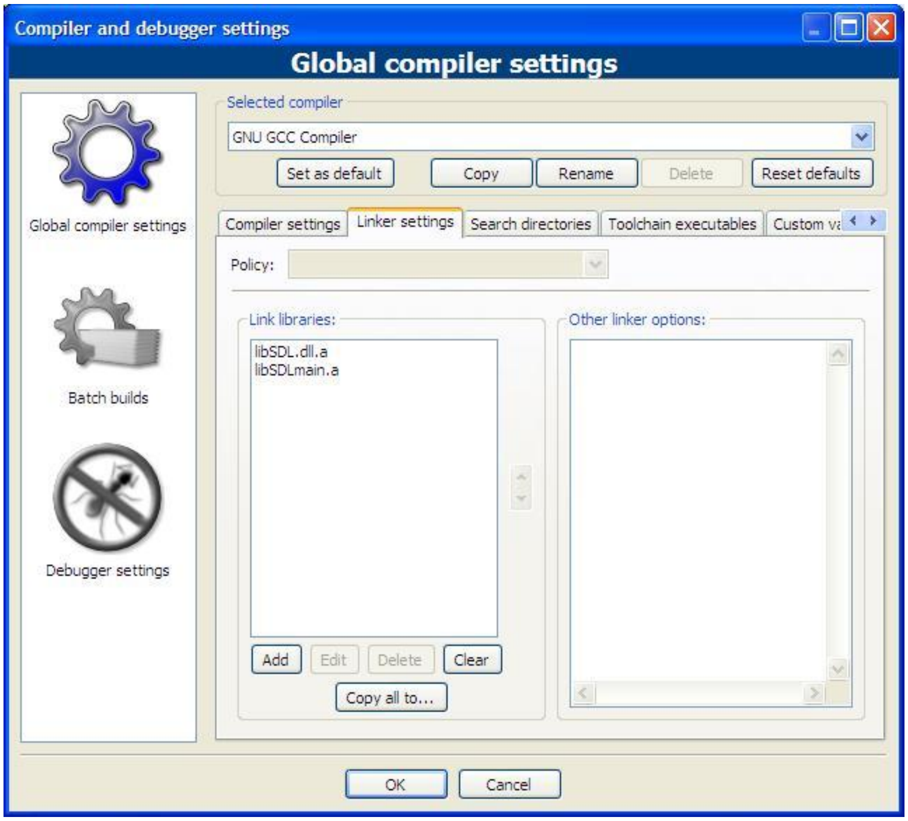
lib: C:\Program Files\CodeBlocks\MinG\

obj:

cflags:

lflags:

ii) Set linker settings
   (1) Add SDL libraries
      (a) Click on “Settings”, “Compiler and Debugger…”. This will open the Compiler and Debugger settings dialog box.
      (b) Click on the “Linker settings” tab.
      (c) Click on “Add” and browse to the lib folder using the same path as in step 2, a, iii. Add “libSDL.dll.a”.
      (d) Click on “Add” again and add “libSDLmain.a”.



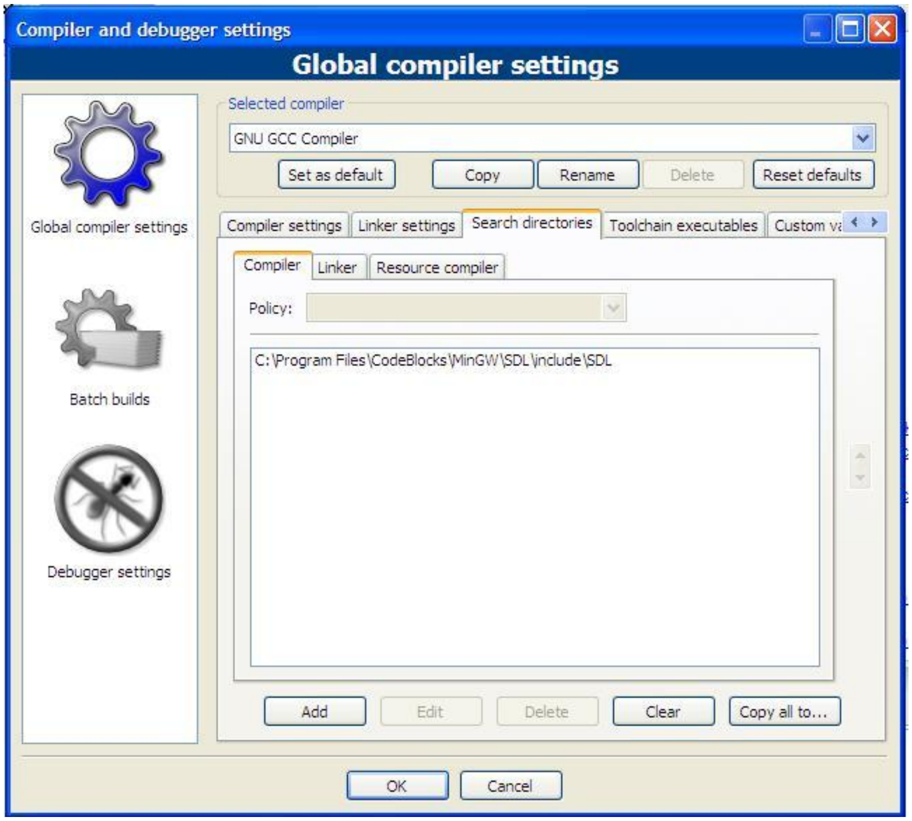
Selected compiler: GNU GCC Compiler

Policy:

Link libraries:
libSDL.dll.a
libSDLmain.a

(2) Add paths to include and lib files

(a) The “Compiler and debugger settings” dialog box should still be open.
(b) Click on the “Search directories” tab.
(c) Click the “Compiler” tab.
(d) Click the “Add” button and add the path to your “include” folder from step 2, 1, ii above, but add /SDL to the end.

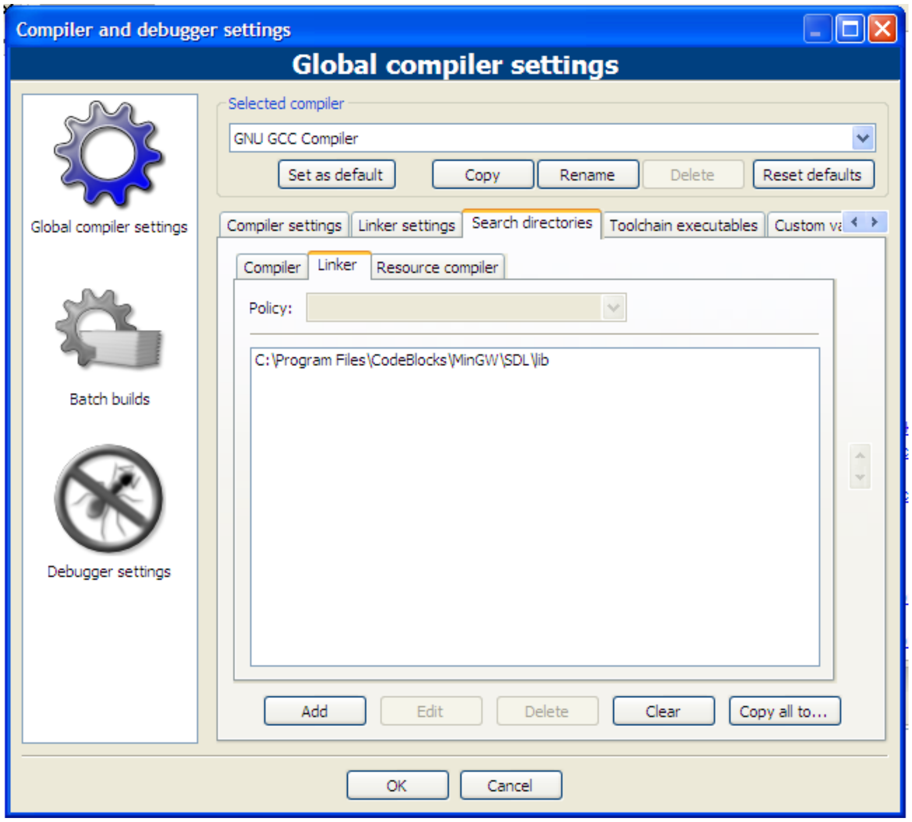


Selected compiler: GNU GCC Compiler

Policy:

C:\Program Files\CodeBlocks\MinGW\SDL\include\SDL

(e) Click the “Linker” tab.
(f) Click the “Add” button. and add the path to your “lib” folder.

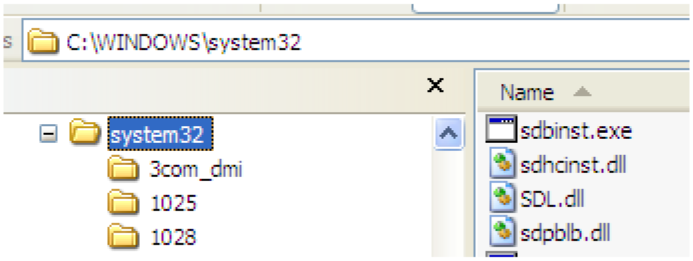


Selected compiler: GNU GCC Compiler

Policy:

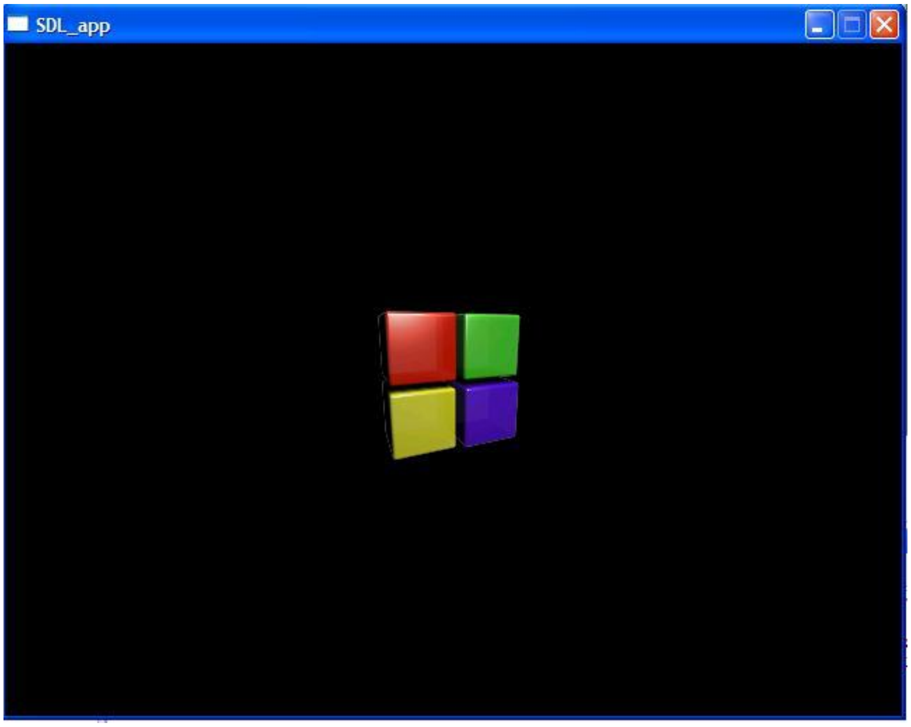
C:\Program Files\CodeBlocks\MinGW\SDL\lib

iii) Add the SDL executable, SDL.dll to your C:\Windows\system32\ folder.



C:\WINDOWS\system32

c) Create an SDL test project
   i) Click on “File”, “New”, “Project” to open the “New from template” dialog box.
   ii) Select “SDL Project”.
   iii) Enter all the usual project properties.
   iv) For the wizard step that asks “Please select SDL’s location” enter the location for the SDL folder that you used in step 2, a, iii.
   v) Compile and run the project. You should see the program display a graphics window with the Code::Blocks logo.



3) Included libraries at LCC
   - a) SDL and SDLmain
     - i) Needed for all SDL programs
     - ii) Available from http://www.libsdl.org
   - b) SDL_mixer
     - i) Sound routines
     - ii) Available from http://www.libsdl.org/projects/SDL_mixer/
   - c) SDL_gfx
     - i) Graphics routines
     - ii) Available from http://www.ferzkopp.net/joomla/software-mainmenu-14/4-ferzkopps-linux-software/19-sdlgfx
   - d) SDL_image
     - i) Image manipulation routines
     - ii) Available from http://www.libsdl.org/projects/SDL_image/