User

Which important c++ libraries are not availavble by default in mingw64 - gcc (g++) - windows version? How to obtain those binaries?

Bing

In the MinGW-w64 environment for Windows, some important C++ libraries may not be available by default. These can include libraries for GUI development, networking, or other specific functionalities that are not part of the standard library. To obtain these libraries, you can use the MSYS2 package manager, which is a part of the MinGW-w64 distribution[**1**](https://stackoverflow.com/questions/1985764/how-to-add-3rd-party-libraries-to-mingw).

Here's a general process to obtain C++ library binaries for MinGW-w64:

1. **Install MSYS2**: If you haven't already, install MSYS2 as it provides a package manager called pacman that you can use to install additional packages.
2. **Use pacman**: With MSYS2 installed, you can use the pacman command to search for and install the libraries you need. For example, to install libxml2, you would use the command pacman -S mingw64/mingw-w64-x86\_64-libxml2[**2**](https://stackoverflow.com/questions/45618405/mingw-64-install-package).
3. **Include Libraries in Your Project**: Once installed, you can include the libraries in your project using the appropriate -l flag when compiling your code.

If you need a specific library that is not available through MSYS2, you may need to download the source code or pre-compiled binaries from the library's official website or a trusted source and manually add them to your project.

Remember to always check the documentation or official repositories of the libraries you're interested in for the most accurate and up-to-date information on how to install and use them with MinGW-w64. If you encounter any issues, the community forums and Stack Overflow are valuable resources for troubleshooting.

Source(s)

1. [How to add 3rd party libraries to MinGW? - Stack Overflow](https://stackoverflow.com/questions/1985764/how-to-add-3rd-party-libraries-to-mingw)

2. [c++ - mingw-64 - Install package - Stack Overflow](https://stackoverflow.com/questions/45618405/mingw-64-install-package)

3. [Cannot build C++ libraries in MinGW - C++ Forum - C++ Users](https://cplusplus.com/forum/beginner/180583/)

4. [c++ - Missing libraries when cross-compiling using mingw - Stack Overflow](https://stackoverflow.com/questions/25233272/missing-libraries-when-cross-compiling-using-mingw)

5. [Get Started with C++ and MinGW-w64 in Visual Studio Code](https://code.visualstudio.com/docs/cpp/config-mingw?source=post_page-----a360be1487c...)

6. [MinGW-w64 - for 32 and 64 bit Windows Wiki2 - SourceForge](https://sourceforge.net/p/mingw-w64/wiki2/Build%20a%20native%20Windows%2064-bit%20gcc%20from%20Linux%20%28including%20cross-compiler%29/)

7. [How do we install new libraries for c++ mingw? - Super User](https://superuser.com/questions/1121913/how-do-we-install-new-libraries-for-c-mingw)

8. [Installing MinGW Tools for C/C++ and Changing ... - GeeksforGeeks](https://www.geeksforgeeks.org/installing-mingw-tools-for-c-c-and-changing-environment-variable/)

9. <https://github.com/libtrading/libtrading%29>

10. <http://repo.msys2.org/msys/x86_64>

11. <https://code.visualstudio.com/docs/cpp/config-mingw?source=post_page-----a360be1487c>

User

You have not mentioned which libraries of c++ are not available by default in mingw64?

Bing

My apologies for the oversight. Some of the C++ libraries that are not included by default in MinGW-w64 for Windows include:

* **OpenBLAS**: A high-performance library for linear algebra operations[**1**](https://superuser.com/questions/1717689/how-can-i-properly-configure-the-g-include-path-with-mingw64).
* **graphics.h**: A header file used for graphics in C++, which is not included in MinGW-w64[**3**](https://superuser.com/questions/1121913/how-do-we-install-new-libraries-for-c-mingw).
* **ncurses**: A library for creating text-based user interfaces[**5**](https://code.visualstudio.com/docs/cpp/config-mingw?source=post_page-----a360be1487c...).

These libraries are not part of the standard C++ library and therefore are not bundled with MinGW-w64. To obtain these libraries, you can use the MSYS2 package manager that comes with MinGW-w64 to install them. For example, to install OpenBLAS, you would use the command pacman -S mingw-w64-x86\_64-openblas in the MSYS2 terminal.

For libraries that are not available through MSYS2, you may need to download the source code or pre-compiled binaries from the library's official website or a trusted source and manually integrate them into your MinGW-w64 environment. Remember to adjust your include paths (-I) and library paths (-L) when compiling your code to link these libraries correctly.

Source(s)

1. [How can I properly configure the g++ include path with mingw64?](https://superuser.com/questions/1717689/how-can-i-properly-configure-the-g-include-path-with-mingw64)

2. [Library <graphics.h> is missing for MinGW-W64 - Stack Overflow](https://stackoverflow.com/questions/72673611/library-graphics-h-is-missing-for-mingw-w64)

3. [How do we install new libraries for c++ mingw? - Super User](https://superuser.com/questions/1121913/how-do-we-install-new-libraries-for-c-mingw)

4. <https://bing.com/search?q=C%2b%2b+libraries+not+included+in+MinGW-w64+by+default>

5. [Get Started with C++ and MinGW-w64 in Visual Studio Code](https://code.visualstudio.com/docs/cpp/config-mingw?source=post_page-----a360be1487c...)

6. <https://stackoverflow.com/questions/12868808/how-to-tell-c-library-path-in-cygwin-and-mingw>

7. <https://stackoverflow.com/questions/4537377/add-extra-include-lib-paths-to-mingw>

8. <https://code.visualstudio.com/docs/cpp/config-mingw?source=post_page-----a360be1487c>

User

Suppose I am not using msys or cygwin but directly using mingw64 using windows path variable from Powershell. How do you I get above libraries. Can I first get them in Msys shell and then copy headers and libraries files from folders inside Msys to respective folders of my msys folders (include, lib, bin, etc.). Can you guide me?

Bing

Certainly! If you're using MinGW-w64 directly through the Windows path variable from PowerShell and not using MSYS2 or Cygwin, you can still obtain the necessary C++ libraries by manually downloading and installing them. Here's a step-by-step guide:

1. **Download the Libraries**: You can download the required libraries from their official websites or trusted sources. Look for pre-compiled binaries for Windows that are compatible with MinGW-w64.
2. **Extract the Files**: Once downloaded, extract the library files to a known location on your system.
3. **Copy to MinGW-w64 Folders**: Copy the header files (.h, .hpp) to the include directory of your MinGW-w64 installation. Similarly, copy the library files (.a, .dll, .lib) to the lib directory, and any binary files (.exe) to the bin directory.
4. **Update the Path Variable**: Ensure that the bin directory of your MinGW-w64 installation is added to the Windows PATH environment variable so that the executables and DLLs can be found by the system.
5. **Link the Libraries**: When compiling your C++ code, link the libraries using the -l flag followed by the name of the library. For example, if you have libxml2.a, you would compile with -llibxml2.
6. **Include Directories**: If the library headers are not in the standard include path, use the -I flag to specify the path to the headers during compilation.

Here's an example of how you might compile a program using the libxml2 library:

g++ -o myprogram myprogram.cpp -llibxml2 -I/path/to/libxml2/include -L/path/to/libxml2/lib

Remember to replace /path/to/libxml2/include and /path/to/libxml2/lib with the actual paths where you copied the header and library files.

It's important to note that some libraries may have dependencies on other libraries, so you'll need to ensure that all dependent libraries are also installed and properly linked.

If you encounter any issues or need specific instructions for a particular library, the community forums and Stack Overflow are valuable resources for troubleshooting and guidance. Additionally, the documentation provided with the library can offer detailed installation and usage instructions.

Source(s)

1. [How can I install MinGW-w64 and MSYS2? - Stack Overflow](https://stackoverflow.com/questions/30069830/how-can-i-install-mingw-w64-and-msys2)

2. [c++ - mingw-64 - Install package - Stack Overflow](https://stackoverflow.com/questions/45618405/mingw-64-install-package)

3. [Install the Mingw and the Cygwin C/C++ compilers under windows](https://medium.com/analytics-vidhya/install-the-mingw-and-the-cygwin-c-c-compilers-under-windows-655b314e644a)

4. [How do we install new libraries for c++ mingw? - Super User](https://superuser.com/questions/1121913/how-do-we-install-new-libraries-for-c-mingw)

5. [Installing MinGW Tools for C/C++ and Changing ... - GeeksforGeeks](https://www.geeksforgeeks.org/installing-mingw-tools-for-c-c-and-changing-environment-variable/)

6. [Get Started with C++ and MinGW-w64 in Visual Studio Code](https://code.visualstudio.com/docs/cpp/config-mingw)

7. [c++ - Many internal std library compile errors when trying to compile ...](https://stackoverflow.com/questions/63291350/many-internal-std-library-compile-errors-when-trying-to-compile-on-msys2-mingw-w)

8. <https://github.com/libtrading/libtrading%29>

9. <http://repo.msys2.org/msys/x86_64>

User

For:

OpenBLAS, graphics.h and ncurses - can you tell me sources where to download them?

Bing

Certainly! Here are the sources where you can download the libraries for OpenBLAS, graphics.h, and ncurses for Windows:

* **OpenBLAS**: You can download OpenBLAS from the official OpenBLAS website or their GitHub repository. They provide binary packages for Windows, which you can find in the Releases section[**1**](https://www.openblas.net/)[**2**](https://github.com/OpenMathLib/OpenBLAS/releases).
* **graphics.h**: This library is not part of the standard MinGW installation. However, you can find a version of graphics.h that is compatible with MinGW at various repositories on GitHub. One such repository provides the necessary files and instructions for setting up graphics.h with Code::Blocks, which you can adapt for use with MinGW[**7**](http://xianyi.github.io/OpenBLAS/).
* **ncurses**: For Windows, you can find ncurses as part of the GnuWin32 project, which provides a setup version of the package with dependencies included, except for msvcrt.dll[**12**](https://stackoverflow.com/questions/63537397/unable-to-use-graphic-h-in-c-in-visual-studio). Alternatively, the official ncurses homepage and the GNU distribution site offer the source code for ncurses[**13**](https://stackoverflow.com/questions/75556484/how-to-install-ncurses-on-windows).

Please note that after downloading these libraries, you will need to manually integrate them into your MinGW-w64 environment by placing the header files in the include directory and the library files in the lib directory. Additionally, ensure that the bin directory is in your Windows PATH environment variable to access any executables or DLLs provided by the libraries. When compiling your code, remember to link the libraries using the appropriate flags..

Source(s)

1. [OpenBLAS : An optimized BLAS library](https://www.openblas.net/)

2. [Releases · OpenMathLib/OpenBLAS · GitHub](https://github.com/OpenMathLib/OpenBLAS/releases)

3. [GitHub - SagarGaniga/Graphics-Library: Download required libraries from ...](https://github.com/SagarGaniga/Graphics-Library)

4. [Ncurses for Windows - GnuWin32](https://gnuwin32.sourceforge.net/packages/ncurses.htm)

5. [Announcing ncurses 6.1](https://www.gnu.org/software/ncurses/)

6. [GitHub - xianyi/openblas/wiki/precompiled-installation-packages](https://github.com/OpenMathLib/OpenBLAS/wiki/Precompiled-installation-packages)

7. [OpenBLAS Homepage - GitHub Pages](http://xianyi.github.io/OpenBLAS/)

8. [OpenBLAS - Browse /v0.2.8 at SourceForge.net](https://sourceforge.net/projects/openblas/files/v0.2.8/)

9. [Library <graphics.h> is missing for MinGW-W64 - Stack Overflow](https://stackoverflow.com/questions/72673611/library-graphics-h-is-missing-for-mingw-w64)

10. [Download graphics.h (graphics.h for MS VS) - SourceForge](https://sourceforge.net/projects/graphicsmsvs/files/0.1/graphics.h/download)

11. [Graphics.h Installation For Code::Blocks Windows version[Updated] - Medium](https://medium.com/@munshisaif/graphics-h-installation-for-code-blocks-windows-version-updated-dedadbcb1ac5)

12. [Unable to use graphic.h in c++ in visual studio - Stack Overflow](https://stackoverflow.com/questions/63537397/unable-to-use-graphic-h-in-c-in-visual-studio)

13. [gcc - how to install ncurses on windows - Stack Overflow](https://stackoverflow.com/questions/75556484/how-to-install-ncurses-on-windows)

14. [Announcing ncurses 6.1](https://www.gnu.org/software/ncurses/ncurses.html)

15. <https://github.com/xianyi/OpenBLAS/releases>

16. <https://github.com/xianyi/OpenBLAS.git>

17. <https://github.com/conda-forge/miniforge>

18. <http://sourceforge.net/projects/openblas/files>

19. <https://www.nuget.org/packages?q=openblas>

20. <https://github.com/Alexpux/MINGW-packages>

21. <http://sourceforge.net/projects/codeblocks/files/Binaries/16.01/Windows/codeblocks-16.01mingw-setup.exe>

22. <https://github.com/SagarGaniga/computer-graphics>

23. <https://github.com/ahuynh359/Graphics>

24. <https://invisible-island.net/archives/ncurses/>

25. <https://invisible-mirror.net/archives/ncurses/>

26. <https://ftp.gnu.org/gnu/ncurses/>

27. [en.wikipedia.org](https://en.wikipedia.org/wiki/Ncurses)

User

Can you tell me pacman packages in Msys for openblas, graphics.h and ncurses?

Bing

Certainly! Here are the pacman packages you can use in MSYS2 for OpenBLAS, graphics.h, and ncurses:

* **OpenBLAS**: You can install the OpenBLAS package using the following command:
* This will install the OpenBLAS library which is an optimized BLAS library based on GotoBLAS2[**2**](https://stackoverflow.com/questions/72673611/library-graphics-h-is-missing-for-mingw-w64).
* **graphics.h**: The graphics.h header file is traditionally associated with the Borland graphics library and is not part of the standard MinGW or MSYS2 installation. However, you can use alternatives like sdl-bgi or WinBGIm, which you would need to compile from source as MSYS2 does not have packages for these[**6**](https://packages.msys2.org/package/mingw-w64-i686-openblas).
* **ncurses**: For the ncurses library, which provides a terminal-based user interface, you can install it using the following command:
* This will install the ncurses library for system V Release 4.0 curses emulation[**11**](https://stackoverflow.com/questions/75556484/how-to-install-ncurses-on-windows).

Please note that the exact package names might vary depending on whether you are using the 32-bit or 64-bit versions of MSYS2, and whether you are using the MinGW or MSYS2 repositories. Make sure to adjust the package names accordingly to your environment.

Source(s)

1. [Base Package: mingw-w64-openblas - MSYS2 Packages](https://packages.msys2.org/base/mingw-w64-openblas)

2. [Library <graphics.h> is missing for MinGW-W64 - Stack Overflow](https://stackoverflow.com/questions/72673611/library-graphics-h-is-missing-for-mingw-w64)

3. [Persuading cmake to find curses in MSYS2? - Stack Overflow](https://stackoverflow.com/questions/67257324/persuading-cmake-to-find-curses-in-msys2)

4. [Package: mingw-w64-x86\_64-openblas - MSYS2 Packages](https://packages.msys2.org/package/mingw-w64-x86_64-openblas)

5. [Package: mingw-w64-clang-x86\_64-openblas - MSYS2 Packages](https://packages.msys2.org/package/mingw-w64-clang-x86_64-openblas)

6. [Package: mingw-w64-i686-openblas - MSYS2 Packages](https://packages.msys2.org/package/mingw-w64-i686-openblas)

7. [Package Management - MSYS2](https://www.msys2.org/docs/package-management/)

8. [pacman package manager - How do I install libpng on MSYS2 ... - Stack ...](https://stackoverflow.com/questions/56613015/how-do-i-install-libpng-on-msys2)

9. [Package: pacman-contrib - MSYS2 Packages](https://packages.msys2.org/package/pacman-contrib?repo=msys&variant=x86_64)

10. [Package: mingw-w64-x86\_64-ncurses - MSYS2 Packages](https://packages.msys2.org/package/mingw-w64-x86_64-ncurses)

11. [gcc - how to install ncurses on windows - Stack Overflow](https://stackoverflow.com/questions/75556484/how-to-install-ncurses-on-windows)

12. <https://www.openblas.net/>

13. <https://github.com/xianyi/OpenBLAS>

14. <https://mirror.msys2.org/mingw/mingw64/mingw-w64-x86_64-openblas-0.3.25-1-any.pkg.tar.zst>

15. <https://mirror.msys2.org/mingw/clang64/mingw-w64-clang-x86_64-openblas-0.3.25-1-any.pkg.tar.zst>

16. <https://mirror.msys2.org/mingw/mingw32/mingw-w64-i686-openblas-0.3.25-1-any.pkg.tar.zst>

17. <https://github.com/Alexpux/MINGW-packages>

18. <https://mirror.msys2.org/msys/x86_64/pacman-contrib-1.9.1-1-x86_64.pkg.tar.zst>

19. <https://www.gnu.org/software/ncurses/>

20. <https://mirror.msys2.org/mingw/mingw64/mingw-w64-x86_64-ncurses-6.4.20230708-1-any.pkg.tar.zst>

User

I downloaded sdl from: https://github.com/genpfault/sdl-bgi in Msys. When I tried command make cross in Msys shell, I got error:

$ make cross for arch in i686-w64-mingw32 x86\_64-w64-mingw32; do \ make install-package arch=$arch prefix=/usr/local/$arch; \ done make[1]: Entering directory '/home/baksh/Downloads/SDL2-2.28.5' \*\*\* ERROR: i686-w64-mingw32 or /usr/local/i686-w64-mingw32 does not exist! make[1]: \*\*\* [Makefile:20: install-package] Error 1 make[1]: Leaving directory '/home/baksh/Downloads/SDL2-2.28.5' make[1]: Entering directory '/home/baksh/Downloads/SDL2-2.28.5' \*\*\* ERROR: x86\_64-w64-mingw32 or /usr/local/x86\_64-w64-mingw32 does not exist! make[1]: \*\*\* [Makefile:20: install-package] Error 1 make[1]: Leaving directory '/home/baksh/Downloads/SDL2-2.28.5' make: \*\*\* [Makefile:15: cross] Error 2

Bing

The error you're encountering indicates that the make command is trying to install packages for the i686-w64-mingw32 and x86\_64-w64-mingw32 architectures, but it cannot find the specified directories. This could be due to several reasons:

1. **The Specified Directories Do Not Exist**: The directories /usr/local/i686-w64-mingw32 and /usr/local/x86\_64-w64-mingw32 may not exist on your system. You might need to create these directories manually or specify the correct paths where your MinGW-w64 toolchain is installed.
2. **Incorrect Makefile Configuration**: The Makefile might be configured to install the packages in directories that do not align with your current setup. You may need to edit the Makefile to point to the correct installation paths for your MinGW-w64 environment.
3. **Using the Wrong Version of Make**: If you're using a version of make that is not compatible with MSYS2, you might encounter errors. Ensure that you have the correct version of make installed that is compatible with MSYS2 and MinGW-w64[**1**](https://stackoverflow.com/questions/51755089/where-is-make-on-msys2-mingw-w64)[**2**](https://stackoverflow.com/questions/45889953/how-to-force-make-to-use-bash-as-a-shell-on-windows-msys2).

To resolve this issue, you can try the following steps:

* **Create the Directories**: If the directories do not exist, create them using the mkdir command:
* **Edit the Makefile**: Modify the Makefile to install the packages in the correct directories where your MinGW-w64 toolchain is located.
* **Install the Correct Version of Make**: If you suspect you're using the wrong version of make, install the correct one using MSYS2's package manager:
* After installing, verify the installation with which make to ensure it points to /usr/bin/make.

If you continue to face issues, you may want to consult the documentation for sdl-bgi or seek assistance from community forums or Stack Overflow, where you can find specific advice for building and installing SDL-related packages on Windows using MinGW-w64 and MSYS2. Remember to provide detailed information about your setup and the steps you've taken so far to help others understand the context of the problem.

Source(s)

1. [Where is make on msys2/mingw-w64? - Stack Overflow](https://stackoverflow.com/questions/51755089/where-is-make-on-msys2-mingw-w64)

2. [How to force make to use bash as a shell on Windows/MSYS2](https://stackoverflow.com/questions/45889953/how-to-force-make-to-use-bash-as-a-shell-on-windows-msys2)

3. [mingw - Using msys shell in Visual Studio Code - Stack Overflow](https://stackoverflow.com/questions/43302853/using-msys-shell-in-visual-studio-code)

4. [MSYS2: Command works fine in terminal, but syntax error `(' in script](https://unix.stackexchange.com/questions/713386/msys2-command-works-fine-in-terminal-but-syntax-error-in-script)

5. <http://gnu.org/licenses/gpl.html>

User

I compiled in SDL in Msys and then copied respective files and folders from lib, bin, include, share, etc folders inside mingw64 folder on my d drive. I want to test sdl. can you give me sample C++ code to test that?

Bing

Certainly! Here's a simple C++ code sample that initializes SDL, creates a window, and then closes it after a keypress. This will help you test if SDL is set up correctly on your system:

#include <SDL.h>

#include <iostream>

int main(int argc, char\* argv[]) {

if (SDL\_Init(SDL\_INIT\_VIDEO) < 0) {

std::cerr << "SDL could not initialize! SDL\_Error: " << SDL\_GetError() << std::endl;

return 1;

}

SDL\_Window\* window = SDL\_CreateWindow("SDL Test Window",

SDL\_WINDOWPOS\_CENTERED,

SDL\_WINDOWPOS\_CENTERED,

640, 480,

SDL\_WINDOW\_SHOWN);

if (window == nullptr) {

std::cerr << "Window could not be created! SDL\_Error: " << SDL\_GetError() << std::endl;

SDL\_Quit();

return 1;

}

SDL\_Event e;

bool quit = false;

while (!quit) {

while (SDL\_PollEvent(&e) != 0) {

if (e.type == SDL\_QUIT || (e.type == SDL\_KEYDOWN && e.key.keysym.sym == SDLK\_ESCAPE)) {

quit = true;

}

}

}

SDL\_DestroyWindow(window);

SDL\_Quit();

return 0;

}

To compile this code, you can use the following command in PowerShell, making sure to replace the paths with the correct ones for your SDL installation:

g++ -o sdl\_test sdl\_test.cpp -ISDL2\include -LSDL2\lib -lSDL2main -lSDL2 -mwindows

This command assumes you have a file named sdl\_test.cpp with the code above. The -ISDL2\include flag tells the compiler where to find the SDL header files, -LSDL2\lib specifies the location of the SDL library files, and -lSDL2main -lSDL2 links against the SDL libraries. The -mwindows flag is used to avoid opening a console window along with the SDL window.

After compiling, you can run the resulting sdl\_test.exe executable to see if a window is created successfully. If everything is set up correctly, you should see a window titled "SDL Test Window" appear on your screen. Pressing the Escape key or closing the window will terminate the program. If you encounter any errors, they will be printed to the standard error stream.

Source(s)

1. [github.com](https://github.com/RealmOfAesir/client/tree/ca46a7c295f071e34140d75a192438592309b159/src%2Fmain.cpp)

User

What is the error in below command. I copied above and created cpp source file. tried to compile:

PS D:\Downloads> g++ -o sdl\_test sdl\_test.cpp -ID:\Programs\mingw64\include\SDL2 D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cchV4Vxg.o:sdl\_test.cpp:(.text+0x1e): undefined reference to SDL\_Init' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cchV4Vxg.o:sdl\_test.cpp:(.text+0x46): undefined reference to SDL\_GetError' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cchV4Vxg.o:sdl\_test.cpp:(.text+0x9d): undefined reference to SDL\_CreateWindow' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cchV4Vxg.o:sdl\_test.cpp:(.text+0xc9): undefined reference to SDL\_GetError' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cchV4Vxg.o:sdl\_test.cpp:(.text+0xeb): undefined reference to SDL\_Quit' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cchV4Vxg.o:sdl\_test.cpp:(.text+0x124): undefined reference to SDL\_PollEvent' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cchV4Vxg.o:sdl\_test.cpp:(.text+0x144): undefined reference to SDL\_DestroyWindow' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cchV4Vxg.o:sdl\_test.cpp:(.text+0x149): undefined reference to SDL\_Quit' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/lib/../lib/libmingw32.a(lib64\_libmingw32\_a-crtexewin.o):crtexewin.c:(.text+0x130): undefined reference to `WinMain' collect2.exe: error: ld returned 1 exit status

Bing

The error you're encountering is a linker error, which occurs because the linker cannot find the SDL library functions that are referenced in your code. This typically happens when the linker flags for the SDL libraries are not specified correctly or are missing entirely.

To resolve this issue, you need to include the linker flags -lSDL2main and -lSDL2 in your g++ command, and you may also need to specify the library path using -L. Additionally, the -mwindows flag is often required when compiling SDL applications on Windows to prevent a console window from appearing alongside your SDL window.

Here's the corrected command:

g++ -o sdl\_test sdl\_test.cpp -ID:\Programs\mingw64\include\SDL2 -LD:\Programs\mingw64\lib -lSDL2main -lSDL2 -mwindows

Make sure to replace D:\Programs\mingw64\lib with the actual path to your SDL2 library files if it's different. The order of the flags can also be important; typically, the -l flags should come after the source files and -I flags in the command[**1**](https://stackoverflow.com/questions/18449697/sdl2-undefined-references-to-functions)[**2**](https://computergraphics.stackexchange.com/questions/8916/sdl2-compiler-throws-undefined-reference)[**3**](https://askubuntu.com/questions/1098073/g-fails-with-an-undefined-reference-to-symbol-dlcloseglibc-2-2-5-even-with)[**4**](https://gamedev.net/forums/topic/655723-undefined-reference-to-sdl_main39/5148450/)[**5**](https://stackoverflow.com/questions/14905239/g-compile-error-undefined-reference-to-a-shared-library-function-which-exists).

If you continue to encounter issues, ensure that the SDL2 library files (.dll, .lib) are present in the specified library path and that the SDL2 development libraries are compatible with MinGW-w64. If necessary, consult the SDL documentation or community forums for additional assistance.

Source(s)

1. [c++ - SDL2 Undefined references to functions - Stack Overflow](https://stackoverflow.com/questions/18449697/sdl2-undefined-references-to-functions)

2. [opengl - sdl2: Compiler throws "Undefined Reference" - Computer ...](https://computergraphics.stackexchange.com/questions/8916/sdl2-compiler-throws-undefined-reference)

3. [compiling - g++ fails with an undefined reference to symbol 'dlclose ...](https://askubuntu.com/questions/1098073/g-fails-with-an-undefined-reference-to-symbol-dlcloseglibc-2-2-5-even-with)

4. [undefined reference to `SDL\_main' - GameDev.net](https://gamedev.net/forums/topic/655723-undefined-reference-to-sdl_main39/5148450/)

5. [g++ compile error: undefined reference to a shared library function ...](https://stackoverflow.com/questions/14905239/g-compile-error-undefined-reference-to-a-shared-library-function-which-exists)

User

Error in compilation with above changes:

D:\Downloads> g++ -o sdl\_test sdl\_test.cpp -ID:\Programs\mingw64\include\SDL2 -LD:\Programs\mingw64\lib -lSDL2main -lSDL2 -mwindows

>>

D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/lib/../lib/libmingw32.a(lib64\_libmingw32\_a-crtexewin.o):crtexewin.c:(.text+0x130): undefined reference to `WinMain' collect2.exe: error: ld returned 1 exit status

Bing

The error message you're seeing indicates that the linker is looking for the WinMain function, which is the entry point for a Windows GUI application. This error can occur if you're trying to compile a program that is expected to be a console application but the -mwindows flag is being used, which tells the linker to look for a Windows GUI entry point.

To resolve this issue, you can try removing the -mwindows flag from your g++ command if your intention is to compile a console application. Here's the updated command without the -mwindows flag:

g++ -o sdl\_test sdl\_test.cpp -ID:\Programs\mingw64\include\SDL2 -LD:\Programs\mingw64\lib -lSDL2main -lSDL2

If your intention is to create a GUI application and you have a WinMain function defined in your code, make sure that it is correctly defined as follows:

int WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PSTR pCmdLine, int nCmdShow) {

// Your code here

}

If you do not have a WinMain function and you are using main instead, you should not use the -mwindows flag, as it is intended for GUI applications with a WinMain entry point.

Additionally, if you are using SDL and want to create a GUI application without a console window, you can define main like this:

int main(int argc, char\* argv[]) {

// Your code here

}

And link with SDL2main which provides a WinMain implementation that calls your main function.

If you continue to face issues, please ensure that the SDL2 development libraries are correctly installed and that the paths in the g++ command are pointing to the correct locations for the include and lib directories. If necessary, consult the SDL documentation or community forums for additional assistance. Remember to provide detailed information about your setup and the steps you've taken so far to help others understand the context of the problem[**1**](https://stackoverflow.com/questions/58324230/undefined-reference-to-winmain-c-mingw)[**2**](https://stackoverflow.com/questions/67514081/i-got-a-error-message-undefined-reference-to-winmain)[**3**](https://bytes.com/topic/c/answers/715077-error-undefined-reference-_winmain-16-when-compiling-cygwin).

Source(s)

1. [Undefined reference to WinMain (C++ MinGW) - Stack Overflow](https://stackoverflow.com/questions/58324230/undefined-reference-to-winmain-c-mingw)

2. [I got a error message "undefined reference to `WinMain'"](https://stackoverflow.com/questions/67514081/i-got-a-error-message-undefined-reference-to-winmain)

3. [ERROR: "undefined reference to `\_WinMain@16'" when compiling in Cygwin ...](https://bytes.com/topic/c/answers/715077-error-undefined-reference-_winmain-16-when-compiling-cygwin)

4. [SFML g++ undefined reference to 'WinMain@16' - Simple and Fast ...](https://en.sfml-dev.org/forums/index.php?topic=25074.0)

5. <https://sourceforge.net/p/mingw-w64/wiki2/Unicode%20apps/>

User

See below result after removing -mwindow:

D:\Downloads> g++ -o sdl\_test sdl\_test.cpp -ID:\Programs\mingw64\include\SDL2 -LD:\Programs\mingw64\lib -lSDL2main -lSDL2 -mwindows

>>

D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/lib/../lib/libmingw32.a(lib64\_libmingw32\_a-crtexewin.o):crtexewin.c:(.text+0x130): undefined reference to WinMain' collect2.exe: error: ld returned 1 exit status PS D:\Downloads> g++ -o sdl\_test sdl\_test.cpp -ID:\Programs\mingw64\include\SDL2 -LD:\Programs\mingw64\lib -lSDL2main -lSDL2 D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/lib/../lib/libmingw32.a(lib64\_libmingw32\_a-crtexewin.o):crtexewin.c:(.text+0x130): undefined reference to WinMain' collect2.exe: error: ld returned 1 exit status

Bing

The error you're encountering, undefined reference to 'WinMain', is a common issue when compiling Windows GUI applications with MinGW. It suggests that the linker is expecting a WinMain function, which is the entry point for Windows GUI applications, instead of the standard main function used in console applications.

To resolve this issue, you can try the following:

1. **Ensure you have a `main` function**: Your application should have a standard main function as the entry point. If you're using SDL, SDL provides its own WinMain that calls your main function.
2. **Remove `-mwindows` flag**: Since you're not creating a GUI application with a WinMain entry point, you should not use the -mwindows flag. This flag tells the linker to look for a WinMain function instead of main.
3. **Link with SDL2 libraries**: Make sure you're linking with the SDL2 libraries correctly. The linker flags -lSDL2main and -lSDL2 should be included at the end of your g++ command.

Here's an example of a corrected g++ command:

g++ -o sdl\_test sdl\_test.cpp -ID:\Programs\mingw64\include\SDL2 -LD:\Programs\mingw64\lib -lSDL2main -lSDL2

If you're still encountering issues, you might want to consider adding the -mconsole flag, which tells the linker to create a console application:

g++ -o sdl\_test sdl\_test.cpp -ID:\Programs\mingw64\include\SDL2 -LD:\Programs\mingw64\lib -lSDL2main -lSDL2 -mconsole

If the problem persists, you can also try using the -municode flag if your application is intended to support Unicode[**1**](https://stackoverflow.com/questions/58324230/undefined-reference-to-winmain-c-mingw). This flag switches to alternate startup code allowing to use wWinMain instead of WinMain.

Please try the above suggestions and let me know if you're able to compile successfully. If you need further assistance, feel free to ask!

Source(s)

1. [Undefined reference to WinMain (C++ MinGW) - Stack Overflow](https://stackoverflow.com/questions/58324230/undefined-reference-to-winmain-c-mingw)

2. [I got a error message "undefined reference to `WinMain'"](https://stackoverflow.com/questions/67514081/i-got-a-error-message-undefined-reference-to-winmain)

3. [c++ - undefined reference to `WinMain@16' - Stack Overflow](https://stackoverflow.com/questions/5259714/undefined-reference-to-winmain16)

4. [undefined reference to WinMain, [Error] Id returned 1 exit status](https://stackoverflow.com/questions/21652922/undefined-reference-to-winmain-error-id-returned-1-exit-status)

5. <https://sourceforge.net/p/mingw-w64/wiki2/Unicode%20apps/>

User

Check the c++ test example code you have provided above. Somewhere, it must be containing reference to winmain:

Error as below:

D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/lib/../lib/libmingw32.a(lib64\_libmingw32\_a-crtexewin.o):crtexewin.c:(.text+0x130): undefined reference to `WinMain' collect2.exe: error: ld returned 1 exit status

Bing

The error you're encountering is related to the linking order of the SDL2 libraries. The linker is expecting a WinMain function because of the -lSDL2main flag, which is used for SDL applications that define main in a specific way expected by SDL. However, the order of the linker flags is crucial for a successful build.

Based on the information from my search results, the correct order of linker flags for compiling an SDL2 application with MinGW should be [-lmingw32 -lSDL2main -lSDL2]([object Object]). This order ensures that the MinGW-specific version of WinMain is used, which then calls your main function.

Here's how you should modify your g++ command:

g++ -o sdl\_test sdl\_test.cpp -ID:\Programs\mingw64\include\SDL2 -LD:\Programs\mingw64\lib -lmingw32 -lSDL2main -lSDL2 -mwindows

Please try this updated command and see if it resolves the issue. If you still encounter problems, ensure that the SDL2 development libraries are correctly installed and that the paths in the g++ command are pointing to the correct locations for the include and lib directories. If necessary, consult the SDL documentation or community forums for additional assistance. Remember to provide detailed information about your setup and the steps you've taken so far to help others understand the context of the problem.

Source(s)

1. [undefined reference to WinMain@16 C++, SDL-2 - Stack Overflow](https://stackoverflow.com/questions/32342285/undefined-reference-to-winmain16-c-sdl-2)

2. <https://bing.com/search?q=SDL2+undefined+reference+to+%60WinMain%60+error>

3. [[sdl2] Undefined reference to WinMain using mingw #25583 - GitHub](https://github.com/microsoft/vcpkg/issues/25583)

4. [Undefined reference to 'WinMain' · Issue #3817 · libsdl-org/SDL](https://github.com/libsdl-org/SDL/issues/3817)

5. [SDL2/FAQWindows - SDL Wiki - Simple DirectMedia Layer](https://wiki.libsdl.org/SDL2/FAQWindows)

6. <https://www.positioniseverything.net/undefined-reference-to-winmain16>

7. <https://www.includehelp.com/c-programs/undefined-reference-to-main-error-in-c.aspx>

8. <https://www.dllme.com/dll/files/sdl2_dll.html>

User

It compiled after above change! Now tell me how to run the code to know whether sdl is working or not?

Bing

Great to hear that the code compiled successfully! To run the compiled SDL application and test if SDL is working, follow these steps:

1. **Open PowerShell or Command Prompt**: Navigate to the directory where your compiled sdl\_test.exe file is located.
2. **Run the Executable**: Type the name of the executable in the command line and press Enter:
3. **Observe the Output**: If SDL is set up correctly, a window titled "SDL Test Window" should appear on your screen. This indicates that SDL is initialized and functioning properly.
4. **Close the Application**: You can close the SDL window by clicking the close button on the window or by pressing the Escape key on your keyboard, as defined in the sample code.

If the window appears without any errors, congratulations, SDL is working on your system! If you encounter any issues while running the application, such as error messages or the window not appearing, make sure that all SDL runtime libraries (DLL files) are available in your system's PATH or in the same directory as the executable.

Feel free to experiment with the SDL library by modifying the sample code to include more SDL features, such as rendering graphics or handling more complex events. Enjoy exploring SDL! 😊

User

Yes, a blank black window titled as above did appear.

Bing

That's fantastic! The appearance of the blank black window titled "SDL Test Window" confirms that SDL is properly installed and working on your system. You've successfully compiled and run a basic SDL application, which is a great starting point for developing more complex graphical applications using SDL.

If you're interested in exploring further, you can start adding more functionality to your application, such as rendering shapes, handling keyboard and mouse input, and creating animations. SDL provides a wide range of features for graphics, sound, and input that you can utilize to create games and other multimedia applications.

If you have any more questions or need further assistance with SDL or C++ programming, feel free to ask. Happy coding! 😊

User

Now after sdl, let's test another library working - openblas. Provide me example code that includes that library with compilation paramaters.

Bing

Certainly! Here's a simple C++ example that uses the OpenBLAS library to perform matrix multiplication using the cblas\_dgemm function from the CBLAS interface:

#include <cblas.h>

#include <iostream>

int main() {

int i = 0;

double A[6] = {1.0, 2.0, 1.0, -3.0, 4.0, -1.0};

double B[6] = {1.0, 2.0, 1.0, -3.0, 4.0, -1.0};

double C[9] = {.5, .5, .5, .5, .5, .5, .5, .5, .5};

cblas\_dgemm(CblasColMajor, CblasNoTrans, CblasTrans, 3, 3, 2, 1, A, 3, B, 3, 2, C, 3);

for (i = 0; i < 9; i++)

std::cout << C[i] << " ";

std::cout << std::endl;

return 0;

}

To compile this code with g++, you will need to link against the OpenBLAS library. Here's an example of how you might compile the code on your system:

g++ -o openblas\_test openblas\_test.cpp -ID:\Programs\mingw64\include\openblas -LD:\Programs\mingw64\lib -lopenblas

Make sure to replace D:\Programs\mingw64\include\openblas and D:\Programs\mingw64\lib with the actual paths to your OpenBLAS include and lib directories, respectively. The -lopenblas flag links against the OpenBLAS library.

After compiling, you can run the resulting openblas\_test.exe executable to see the output of the matrix multiplication. If everything is set up correctly, you should see the results printed to the console[**2**](https://stackoverflow.com/questions/38056069/openblas-and-g)[**3**](https://stackoverflow.com/questions/30035531/how-to-link-c-to-openblas).

If you encounter any issues during compilation or execution, make sure that the OpenBLAS development libraries are correctly installed and that the paths in the g++ command are pointing to the correct locations for the include and lib directories. If necessary, consult the OpenBLAS documentation or community forums for additional assistance.

Source(s)

1. [User Manual · OpenMathLib/OpenBLAS Wiki · GitHub](https://github.com/OpenMathLib/OpenBLAS/wiki/User-Manual)

2. [c++ - OpenBlas and g++ - Stack Overflow](https://stackoverflow.com/questions/38056069/openblas-and-g)

3. [linker - How to link c++ to OpenBlas? - Stack Overflow](https://stackoverflow.com/questions/30035531/how-to-link-c-to-openblas)

4. [Using OpenBLAS library - IBM - United States](https://www.ibm.com/docs/en/open-xl-c-cpp-zos/1.1?topic=libraries-using-openblas-library)

5. [bgeneto/build-install-compile-openblas - GitHub](https://github.com/bgeneto/build-install-compile-openblas)

6. [c - Correct usage of cblas\_dgemm - Stack Overflow](https://stackoverflow.com/questions/71369489/correct-usage-of-cblas-dgemm)

7. [OpenBLAS – High Performance Computing - NTNU](https://www.hpc.ntnu.no/idun/software-on-idun/openblas/)

8. <https://github.com/xianyi/OpenBLAS/issues/936>

9. <https://gist.github.com/xianyi/5780018>

10. <https://gist.github.com/xianyi/6930656>

11. [github.com](https://github.com/liweijia/TensorSloW/tree/0dfc56806595db33ddb2d2c4a06e6cbebd5f81d5/aux%2Ftest_cblas_dgemm.c)

12. [github.com](https://github.com/frankwillmore/spack-alcf/tree/af4a36c4d99d7953200d369bbeba6303ad87dcf7/var%2Fspack%2Frepos%2Fbuiltin%2Fpackages%2Fatlas%2Ftest_cblas_dgemm.c)

13. [github.com](https://github.com/rostam/Experiments/tree/80e204939a74ceb6e2f0317e00f0f8cd22fe0fa4/cpp%2Fcpp-new-features%2Fblas_lapack.cpp)

User

To include comments shall I put // in the beginning of the lines?

Bing

Yes, in C++ you can include comments by putting // at the beginning of the line for single-line comments. Anything following // on that line will be considered a comment and will not be executed as part of the program.

Here's an example:

#include <iostream>

int main() {

// This is a single-line comment

std::cout << "Hello, World!" << std::endl;

// The line below is commented out and will not execute

// std::cout << "This line will not print!" << std::endl;

return 0;

}

For multi-line comments, you can use /\* to start the comment block and \*/ to end it. Here's how you can do it:

#include <iostream>

int main() {

/\* This is a multi-line comment.

It can span multiple lines.

The compiler will ignore these lines. \*/

std::cout << "Hello, World!" << std::endl;

return 0;

}

Remember that comments are a great way to document your code and make it easier for others (and yourself) to understand what the code is supposed to do. They are also useful for temporarily disabling code during debugging. If you have any more questions or need further assistance with C++ programming, feel free to ask!

User

Openblas test worked flawlessly! Now ncurses test. Can you give me example code and compilation command with arguments? Also whether ncurses is purely c++ library or also used in C?

Bing

I'm glad to hear that the OpenBLAS test worked well! Now, let's move on to testing ncurses.

Ncurses is a library that provides functions to write text-based user interfaces in a terminal-independent manner. It is used in both C and C++ programming[**1**](https://www.codedrome.com/an-introduction-to-ncurses-in-c/).

Here's a simple example code in C++ that uses ncurses to display "Hello, World!" on the screen:

#include <ncurses.h>

int main() {

initscr(); // Initialize the window

printw("Hello, World!"); // Print Hello, World

refresh(); // Print it on the real screen

getch(); // Wait for user input

endwin(); // End curses mode

return 0;

}

To compile this code with g++, you will need to link against the ncurses library. Here's the compilation command with arguments:

g++ -o ncurses\_test ncurses\_test.cpp -lncurses

This command will compile the source file ncurses\_test.cpp and link it with the ncurses library, producing an executable named ncurses\_test.

After compiling, you can run the resulting ncurses\_test.exe executable to see the "Hello, World!" message displayed in your terminal window. If everything is set up correctly, you should be able to interact with the program as expected[**7**](https://stackoverflow.com/questions/66063363/how-to-compile-ncurses-program-for-native-windows-use)[**9**](https://tldp.org/HOWTO/NCURSES-Programming-HOWTO/helloworld.html).

If you encounter any issues during compilation or execution, make sure that the ncurses development libraries are correctly installed and that the paths in the g++ command are pointing to the correct locations for the include and lib directories. If necessary, consult the ncurses documentation or community forums for additional assistance.

Source(s)

1. [An Introduction to ncurses in C - CodeDromeCodeDrome](https://www.codedrome.com/an-introduction-to-ncurses-in-c/)

2. [Introduction to Ncurses (Part 1) - DEV Community](https://dev.to/tbhaxor/introduction-to-ncurses-part-1-1bk5)

3. [quick&dirty example on how to use ncurses library in c++ to build a ...](https://gist.github.com/sebcode/939869)

4. [ncurses - how to properly use curses/debugging a simple c program ...](https://stackoverflow.com/questions/13371869/how-to-properly-use-curses-debugging-a-simple-c-program)

5. [Game Programming in C with the Ncurses Library | Viget](https://www.viget.com/articles/game-programming-in-c-with-the-ncurses-library/)

6. [Does including ncurses.h in C programming also include stdio.h?](https://stackoverflow.com/questions/60291027/does-including-ncurses-h-in-c-programming-also-include-stdio-h)

7. [How to Compile Ncurses Program for Native Windows Use](https://stackoverflow.com/questions/66063363/how-to-compile-ncurses-program-for-native-windows-use)

8. [Not able to compile C/C++ code that's using ncurses](https://stackoverflow.com/questions/6538059/not-able-to-compile-c-c-code-thats-using-ncurses)

9. [Hello World - Linux Documentation Project](https://tldp.org/HOWTO/NCURSES-Programming-HOWTO/helloworld.html)

10. [GitHub - dafrito/NCurses-Examples: Small collection of C++ examples ...](https://github.com/dafrito/NCurses-Examples)

11. [A general purpose example of using ncurses in C++ e.g. with STL strings ...](https://gist.github.com/daleobrien/7535489)

User

ncurses compilation was not successful. I tried -I flag also. All outputs you can see below:

PS D:\Downloads> g++ -o ncurses\_test ncurses\_test.cpp -lncurses ncurses\_test.cpp:1:10: fatal error: ncurses.h: No such file or directory 1 | #include <ncurses.h> | ^~~~~~~~~~~ compilation terminated.

PS D:\Downloads> g++ -o ncurses\_test ncurses\_test.cpp -ID:\Programs\mingw64\include\ncursesw -lncurses D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\ccUDKOgR.o:ncurses\_test.c:(.text+0x10): undefined reference to \_\_imp\_initscr' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\ccUDKOgR.o:ncurses\_test.c:(.text+0x23): undefined reference to \_\_imp\_printw' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\ccUDKOgR.o:ncurses\_test.c:(.text+0x2c): undefined reference to \_\_imp\_refresh' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\ccUDKOgR.o:ncurses\_test.c:(.text+0x35): undefined reference to \_\_imp\_stdscr' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\ccUDKOgR.o:ncurses\_test.c:(.text+0x42): undefined reference to \_\_imp\_wgetch' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\ccUDKOgR.o:ncurses\_test.c:(.text+0x4b): undefined reference to \_\_imp\_endwin' collect2.exe: error: ld returned 1 exit status

PS D:\Downloads> g++ -o ncurses\_test ncurses\_test.cpp -ID:\Programs\mingw64\include\ncurses -lncurses D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cc3cjDEn.o:ncurses\_test.c:(.text+0x10): undefined reference to \_\_imp\_initscr' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cc3cjDEn.o:ncurses\_test.c:(.text+0x23): undefined reference to \_\_imp\_printw' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cc3cjDEn.o:ncurses\_test.c:(.text+0x2c): undefined reference to \_\_imp\_refresh' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cc3cjDEn.o:ncurses\_test.c:(.text+0x35): undefined reference to \_\_imp\_stdscr' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cc3cjDEn.o:ncurses\_test.c:(.text+0x42): undefined reference to \_\_imp\_wgetch' D:/Programs/mingw64/bin/../lib/gcc/x86\_64-w64-mingw32/13.2.0/../../../../x86\_64-w64-mingw32/bin/ld.exe: C:\Users\baksh\AppData\Local\Temp\cc3cjDEn.o:ncurses\_test.c:(.text+0x4b): undefined reference to \_\_imp\_endwin' collect2.exe: error: ld returned 1 exit status

Bing

It seems you're encountering issues with finding the ncurses.h header file and linking against the ncurses library. The errors indicate that the compiler cannot find the header file, and the linker cannot find the ncurses functions.

To resolve the compilation issue, ensure that you're including the correct path to the ncurses header files using the -I flag. If you're using MinGW-w64 and the ncurses header files are located in D:\Programs\mingw64\include\ncursesw, your include path should be correct. However, if the header files are in a different directory, you'll need to adjust the -I flag accordingly.

For the linking errors, it's possible that you need to link against the ncursesw (wide-character support) library instead of ncurses. You can try using -lncursesw instead of -lncurses in your g++ command. Additionally, you may need to define NCURSES\_STATIC if you're statically linking against ncurses[**3**](https://stackoverflow.com/questions/56504549/unable-to-compile-program-with-ncurses-h-header-using-mingw).

Here's an updated g++ command that includes the NCURSES\_STATIC definition and links against the ncursesw library:

g++ -o ncurses\_test ncurses\_test.cpp -ID:\Programs\mingw64\include\ncursesw -DNCURSES\_STATIC -lncursesw

If you continue to face issues, you might want to consider using PDCurses, which is a port of the ncurses library for the Windows platform and might be easier to set up and use in a MinGW environment[**2**](https://stackoverflow.com/questions/66063363/how-to-compile-ncurses-program-for-native-windows-use).

Please try the updated command and let me know if it resolves the issue. If you need further assistance, feel free to ask!

Source(s)

1. [Cannot static link ncurses with MinGW - Stack Overflow](https://stackoverflow.com/questions/67656787/cannot-static-link-ncurses-with-mingw)

2. [How to Compile Ncurses Program for Native Windows Use](https://stackoverflow.com/questions/66063363/how-to-compile-ncurses-program-for-native-windows-use)

3. [Unable to compile program with <ncurses.h> header using MinGW](https://stackoverflow.com/questions/56504549/unable-to-compile-program-with-ncurses-h-header-using-mingw)

4. <https://github.com/aviikc/CIS2500_fileeditor>

User

The above worked. When I ran, a window opened with hello world message but on closing that window, powershell was also exited. I want hello world to open in another window.

Bing

I'm glad to hear that the ncurses test worked! The behavior you're experiencing is typical when running console applications that create new windows. When you close the ncurses window, it also closes the associated console window because they are linked.

To open the ncurses window in a separate console window, you can run your ncurses\_test.exe from a separate instance of the command prompt or PowerShell. Here's how you can do it:

1. **Create a Shortcut**: Right-click on your desktop or in a folder, select "New," and then "Shortcut."
2. **Set the Location**: In the shortcut creation wizard, set the location to the path of your ncurses\_test.exe executable. For example:
3. **Finish the Setup**: Give the shortcut a name and finish setting it up.
4. **Run the Shortcut**: Double-click the shortcut to run your ncurses\_test.exe in a new window.

By running the executable via a shortcut, you should be able to close the ncurses window without closing the original PowerShell window from which you launched the shortcut.

Alternatively, you can open a new instance of PowerShell or Command Prompt, navigate to the directory where your executable is located, and run it from there. This way, closing the ncurses window will only close the new instance, not the original one.

If you have any more questions or need further assistance, feel free to ask!

User

Is it not possible to define your test code in such a way that it opens a new window?

Bing

The ncurses library is designed to work within the terminal window and does not create separate graphical windows like GUI libraries such as SDL or Qt. When you run an ncurses application, it takes over the terminal window to provide a text-based user interface.

If you want to keep your PowerShell window open after running the ncurses application, you can run the executable in a new PowerShell window. Here's how you can do it:

1. **Open a New PowerShell Window**: Right-click on the PowerShell icon and select "Run as Administrator" or simply "PowerShell" to open a new instance.
2. **Navigate to Your Executable**: Use the cd command to change directories to where your ncurses\_test.exe is located.
3. **Run Your Executable**: Type .\ncurses\_test.exe and press Enter to run the application.

This way, when you close the ncurses application, it will only close the new PowerShell window, and your original PowerShell window will remain open.

If you're looking for a way to create a completely separate GUI window, you would need to use a different library that is designed for graphical user interface development, such as Qt or wxWidgets, which are capable of creating standalone windows independent of the console.

If you have any more questions or need further assistance, feel free to ask!

User

Can you combine ncurses with sdl (which we saw earlier) in the test code to create separate window?

Bing

Combining ncurses and SDL in the same application to create separate windows is not a typical use case, as ncurses is designed for text-based terminal interfaces and SDL is used for creating graphical windows. However, it is possible to run both libraries in the same application, but they will not interact directly with each other; each will manage its own window.

According to the information I found, there is no built-in support in ncurses for embedding an SDL graphical context within an ncurses window[**1**](https://stackoverflow.com/questions/17702353/ncurses-with-graphics-window). If you want to use both, you would typically have two separate parts of your application: one part using ncurses for text-based interaction in the terminal, and another using SDL for graphical output in a separate window.

If you're looking to create a text-based interface within an SDL window, you might consider using a library like PDCurses, which is a port of ncurses that can work with SDL[**2**](https://pdcurses.org/sdl2/). This allows you to use curses-style functions to manage text within an SDL application, effectively giving you the ability to create a text UI within a graphical window.

Here's a conceptual example of how you might structure an application that uses both SDL and ncurses:

#include <SDL.h>

#include <ncurses.h>

#include <thread>

void run\_ncurses() {

initscr();

printw("Hello from ncurses!");

refresh();

getch();

endwin();

}

void run\_sdl() {

SDL\_Init(SDL\_INIT\_VIDEO);

SDL\_Window\* window = SDL\_CreateWindow("SDL Window", SDL\_WINDOWPOS\_CENTERED, SDL\_WINDOWPOS\_CENTERED, 640, 480, SDL\_WINDOW\_SHOWN);

SDL\_Renderer\* renderer = SDL\_CreateRenderer(window, -1, SDL\_RENDERER\_ACCELERATED);

// Your SDL rendering loop here

SDL\_DestroyRenderer(renderer);

SDL\_DestroyWindow(window);

SDL\_Quit();

}

int main() {

std::thread ncurses\_thread(run\_ncurses);

std::thread sdl\_thread(run\_sdl);

ncurses\_thread.join();

sdl\_thread.join();

return 0;

}

In this example, we create two separate threads: one for running the ncurses part of the application and another for the SDL part. Each thread manages its own window independently.

To compile this code, you would need to link against both the ncurses and SDL libraries:

g++ -o combined\_test combined\_test.cpp -lncurses -lSDL2 -lSDL2main -pthread

Please note that this is a conceptual example and may require additional setup and error handling to work correctly in your environment. Also, mixing two libraries that manage their own windows can lead to complex issues, so it's important to carefully manage the lifecycle of each library within your application.

If you have any more questions or need further assistance, feel free to ask!

Source(s)

1. [sdl - ncurses with graphics window - Stack Overflow](https://stackoverflow.com/questions/17702353/ncurses-with-graphics-window)

2. [PDCurses for SDL 2.x | PDCurses](https://pdcurses.org/sdl2/)

3. [Is there an SDL emulation of how ncurses works? : r/roguelikedev - Reddit](https://www.reddit.com/r/roguelikedev/comments/98ly18/is_there_an_sdl_emulation_of_how_ncurses_works/)

User

Okay forget about sdl and ncurses combine. As stated earlier can you give a code that combines QT and ncurses so that both can use console and powershell is also not exited? Also which packages I have to install in Msys to enable QT?

Bing

Combining Qt and ncurses in the same application is not a common practice, as Qt is primarily used for graphical user interfaces, while ncurses is used for text-based terminal interfaces. However, it is possible to use Qt for the core application logic and ncurses for a text-based user interface if desired[**4**](https://packages.msys2.org/package/mingw-w64-x86_64-qt-creator?repo=mingw64).

To enable Qt in MSYS2, you will need to install the Qt packages. You can do this by opening the MSYS2 terminal and running the following command:

pacman -S mingw-w64-x86\_64-qt5

This command will install the Qt5 packages for the 64-bit MinGW-w64 environment. If you need other Qt components or a different version, you can search for available Qt packages using:

pacman -Ss qt5

After installing Qt, you can write a Qt application that uses the core and SQL modules, for example, and use a QTimer to refresh the interface or perform other timed actions. For the ncurses part, you would typically run it in a separate thread or process to avoid blocking the Qt event loop[**4**](https://packages.msys2.org/package/mingw-w64-x86_64-qt-creator?repo=mingw64).

Here's a conceptual example of how you might structure a Qt application that also uses ncurses:

#include <QCoreApplication>

#include <QTimer>

#include <iostream>

#include <thread>

#include <ncurses.h>

void run\_ncurses() {

initscr(); // Initialize the window

printw("Hello from ncurses!"); // Print Hello, World

refresh(); // Print it on the real screen

getch(); // Wait for user input

endwin(); // End curses mode

}

int main(int argc, char \*argv[]) {

QCoreApplication app(argc, argv);

// Set up a QTimer to perform actions in the Qt part of the application

QTimer timer;

QObject::connect(&timer, &QTimer::timeout, []() {

// Perform timed actions here

std::cout << "Qt timer tick" << std::endl;

});

timer.start(1000); // Set the timer to tick every second

// Run ncurses in a separate thread

std::thread ncurses\_thread(run\_ncurses);

// Start the Qt event loop

int result = app.exec();

// Wait for the ncurses thread to finish

ncurses\_thread.join();

return result;

}

To compile this code, you would need to link against both the Qt and ncurses libraries. Make sure to have the Qt environment set up correctly in your MSYS2 environment before compiling.

Please note that this is a simplified example and may require additional setup and error handling to work correctly in your environment. If you have any more questions or need further assistance, feel free to ask!

Source(s)

1. [c++ - Ncurses and Qt Interoperability - Stack Overflow](https://stackoverflow.com/questions/20606318/ncurses-and-qt-interoperability)

2. [MSYS2 - Qt Wiki](https://wiki.qt.io/MSYS2)

3. [Packages - MSYS2 Packages](https://packages.msys2.org/package/)

4. [Package: mingw-w64-x86\_64-qt-creator - MSYS2 Packages](https://packages.msys2.org/package/mingw-w64-x86_64-qt-creator?repo=mingw64)

5. [GitHub - libyui/libyui: Libyui is a widget abstraction library ...](https://github.com/libyui/libyui)

6. <https://github.com/KubaO/stackoverflown/tree/master/questions/ncurses-20606318>

7. <https://www.qt.io/>

8. <https://mirror.msys2.org/mingw/mingw64/mingw-w64-x86_64-qt-creator-12.0.0-1-any.pkg.tar.zst>

User

Okay so QT also uses GUI. Is there no option where I can combine with ncurses with another library so that execution is done in a separate window?

Bing

Ncurses is specifically designed for creating text-based interfaces within terminal windows, and it doesn't inherently support creating separate graphical windows. If you want to create a separate window for graphical output while using ncurses for the terminal interface, you would typically use a GUI library like SDL, Qt, or GTK+ for the graphical part of your application.

However, if you want to keep the terminal window open and separate from the GUI window, you can consider the following approaches:

1. **Separate Processes**: Run the ncurses interface and the GUI interface as separate processes. You can use inter-process communication (IPC) mechanisms to communicate between the two if needed.
2. **Threads**: Use threads to run the ncurses interface and the GUI interface concurrently within the same application. This requires careful synchronization but allows both interfaces to operate independently.
3. **Terminal Emulation in GUI**: Some GUI libraries offer terminal emulation widgets that can mimic the behavior of terminal interfaces within a graphical window. For example, GTK+ has the VTE widget, and Qt has QTermWidget.

For the GUI part, you can install the necessary packages in MSYS2 using the pacman package manager. For example, to install GTK+, you can use:

pacman -S mingw-w64-x86\_64-gtk3

And for Qt, as mentioned earlier:

pacman -S mingw-w64-x86\_64-qt5

Remember to choose the library that best suits your application's needs and consider the complexity of managing both a text-based and a graphical interface within the same application. If you have any more questions or need further assistance, feel free to ask!

Source(s)

1. [How to Compile Ncurses Program for Native Windows Use](https://stackoverflow.com/questions/66063363/how-to-compile-ncurses-program-for-native-windows-use)

2. [Announcing ncurses 6.1](https://www.gnu.org/software/ncurses/ncurses.html)

3. [Introduction to Ncurses (Part 1) - DEV Community](https://dev.to/tbhaxor/introduction-to-ncurses-part-1-1bk5)

User

I tested ncurses code as above. It opened a new terminal windows. Then I pressed backspace instead of enter and it returned to Powershell! no powershell exit issue! You could have guided me with this simple trick instead of directing to more complicated options as above.

Bing

I apologize for any confusion caused. It's great to hear that you found a simple solution that works for your needs. Using the backspace key to exit the ncurses window and return to PowerShell without closing the PowerShell window is indeed a straightforward approach.

If you have any more questions or if there's anything else I can assist you with, please let me know. I'm here to help! 😊