Set up Code::Blocks on Windows

April 13, 2013

Set up Code::Blocks on Windows

In order to be able to compile our code on Windows, it is necessary to properly set up Code::Blocks (a cross-platform IDE) to use Windows compiler. It is also needed to obtain some external libraries to get the code to compile successfully. Below are briefly reported all the required steps. Each of them will be explained in further details.

Required steps

- Install Visual Studio Express 2010
- Install Windows SDK
- Install Code::Blocks
- Set up Code::Blocks to work with MSVS compiler
- Get and link against Gsl libraries
- Get and link against Boost libraries

Install Visual Studio Express 2010

Visual Studio Express can be downloaded free of charge directly from Microsoft Website. It is not strictly required to set up Code::Blocks, but it includes the compiler and, above all, it will be useful to compile libraries. The installation should be straightforward.

Install Windows SDK

Windows SDK (Software Development Kit) is needed in order to properly set up Code::Blocks on Windows. The SDK can be downloaded from Microsoft Website depending on your own version of Windows. It is available both as a ISO image and a web installer. This installation could present some problems. Some of them are known issues and have a known solution: Most likely the installation will fail and a totally useless error message will be displayed, such as *A problem occurred while installing selected Windows SDK components.* If it occurs, it is necessary to open the log file and search for an error code or return status.

error 5100: Manually remove any "Microsoft Visual C++ 2010 Redistributable" (both x86 and x64) and run the installation again;

error 1603: Remove any SDK installation that might appear and run the installation again unchecking "Visual c++ compiler" on the options page.

Install Code::Blocks

Code::Blocks is a cross-platform IDE (Integrated Development environment). The choice of this IDE is due to the possibility to use it both on Windows and Linux, eliminating the necessity of handling different projects depending on the used platform.

Code::Blocks is an open-source IDE and a Windows installer can be downloaded from the official website. The installation should not present any problem.

Set up Code::Blocks

On its first startup, Code::Blocks should detect already installed compilers. If Visual c++ compiler is not automatically detected, you'll have to configure it manually. Here is a brief guide to how to do it:

Settings → Compiler: select "Microsoft Visual C++ 2010" from the drop-down list. Select the "Toolchain executables" tab. The Compiler's installation directory field must be filled with the path of the folder which contains the folder bin where the compiler is. For example: *C: \Program Files (x86) \Microsoft Visual Studio 10.0 \VC*.

In the "Program Files" tab, the following fields should be filled as below:

C compiler cl.exe

C++ compiler cl.exe

Linker for dynamic libs link.exe

Linker for static libs link.exe

Debugger cdb (default)

Resource compiler rc.exe

Make program nmake.exe

In the "Additional Path" tab, the following entries have to be added (They might need to be specialized for your own installation directory):

C:\Program Files (x86)\Microsoft Visual Studio 10.0\Common7\IDE
C:\Program Files (x86)\Microsoft SDKs\Windows\v7.0A
C:\Program Files (x86)\Microsoft SDKs\Windows\v7.0A\Bin

Now switch to the "Search directories" tab. Both in the "Compiler" tab and "Resource compiler" tab add the following entries:

C:\Program Files (x86)\Microsoft Visual Studio 10.0\VC\include
C:\Program Files (x86)\Microsoft SDKs\Windows\v7.0A\Include

In the "Linker" tab add:

C:\Program Files (x86)\Microsoft Visual Studio 10.0\VC\lib
C:\Program Files (x86)\Microsoft SDKs\Windows\v7.0A\Lib

Now browse to the 'Compiler setting" tab and, in the "Other options" tab, add: *\EHsc*. This can be done either here or in the same tab accessed by right-clicking on the project and selecting "Build options".

Gsl Libraries[1]

GSL (GNU Scientific Library) is a c++ library released under the GNU General Public Licence. On the internet can be found some precompiled libraries for Windows, but one can download the source codes and compile them himself. Below there is a brief guide to how to do it.

First, download the source codes from the official website (last version at the time of writing is gsl-1.15.tar.gz). Unpack the archives until you get a gsl-1.15 folder which will be assumed to be your root directory from now on.

Download the gsl-1.15.vc10.zip archives and unpack it in the root directory. This should add a folder named 'build.vc10'. During the unpacking, overwrite any file you are asked for.

[1]This guide is an extract from the Brian Gladman's guide to build GSL libraries on Windows, which can be found at this url: `http://svn.icmb.utexas.edu/svn/repository/trunk/zpub/sdkpub/gsl-1.15/vs2010_port/gsl.vc10.readme.txt`

Open the solution file *gsl.lib.sln* with Visual Studio 2010. Build only the *gslhdrs* project and run the resulting exe.

Now run the *gsllib* and *cblaslib* projects in both Debug and Release mode.

Repeat last two steps for the *gsl.dll.sln* solution to build dll libraries. Resulting built libraries will be placed respectively in the following directories under proper subdirectories:
gsl-1.15\build.vc10\lib\
gsl-1.15\build.vc10\dll\

It could happen that some errors prevent the building of the libraries. Below there is a list of workarounds for known issues:
In the *rk4imp.c* file, replace any occurrence of sqrt(3) with M_SQRT3 In the *bspline.c* file, move variables declarations to the first part of the function: gsl_bspline_greville_abscissa(size_t i, gsl_bspline_workspace *w)

Once the libraries have been built, Code::Blockshas to be set up to link against them. This can be done either in the general compiler setting or in the project's build options. Open the Build options of the project by right-clicking on it and browse to the Compiler Search Directories tab and add the path to the gsl-1.15 folder (the one we previously named the root directory). Then browse to the Linker settings tab and, in the Link libraries filed, add the path to the *gsl.lib* and *cblas.lib* (they should be inside the \build.vc10\lib folder under the proper subfolder)

Boost libraries[2]

As we did with GSL libraries, below there is a guide to how to compile Boost libraries for Windows.
Go to the directory tools\build\v2\

Run bootstrap.bat

Run b2 install –prefix=PREFIX where PREFIX is the directory where you want Boost.Build to be installed

Add PREFIX\bin to your PATH environment variable.

[2]This guide is an extract from the Getting Started guide of the official Boost website

Change your current directory to the Boost root directory and invoke b2 as follows:

b2 –build-dir=buil-directory toolset=toolset-name –build-type=complete stage

Once the libraries have been built, Code::Blockshas to be set up to link against them. Right-click on the project and go to "Build options".

In the "Linker settings" tab add the path to the program options library: *libboost_program_options-vc100-mt-s-1_53.lib*

In the "Search directories" tab, "compiler" tab, add the path to the folder *boost_1_53_0*; in the "linker" tab add the path to *boost_1_53_0\lib* and *boost_1_53_0\bin.v2\libs*