

DEVELOPING SCL PROGRAMS (C)

On Windows and Linux

Using the Command Prompt and Codeblocks

Dr. José M. Garrido
Department of Computer Science

February 2020

College of Computing and Software Engineering
Kennesaw State University

© 2020, J. M. Garrido

1 Introduction

Software can be developed using directly the operating systems shell commands with an appropriate editor, or an Integrated Development Environment (IDE). There are several IDEs that help developing programs written in C, C++, Fortran, and other programming languages. Eclipse is one of the most complete and powerful tools, however it is mainly useful for implementing programs in languages such as Java and Python and in general, it can be very slow. For SCL, C++, C, and Fortran programs, Codeblocks and CodeLite are faster, lighter, and easier tools to use.

This document briefly explains developing SCL programs and translation to C using:

- The Linux and Windows commands necessary that can be used on a Terminal or Command window.
- CodeBlocks on Linux and Windows

Note that for Windows, there two versions of the SCL compiler/translator:

- The first one, `sclv2.exe`, built with MinGW GNU C++ compiler tools.
- The second one, `sclv2vs.exe`, built with Visual Studio C++ (Community).

2 Downloading and Installing the SCL Files

2.1 Linux

1. On Linux, download the archive file: `SCL_linux.tar.gz` from the following web page.

`ksuweb.kennesaw.edu/~jgarrido/SCL_programs`

2. Select the appropriate archive and extract all files from the archive.
3. The compiler/translator file (`sclv2.out`), the various script files, sample SCL programs must be stored on a folder such as `~/SCL_progs`.
4. The subdirectory `progs`, contains the sample SCL programs and have the `scl` extension.

2.2 Windows

1. On Windows, the name of the archive is **SCL.Win.zip**. As mentioned previously, there two executable files of the SCL compiler/translator. The first one is the file **sclv2_Mingw.exe** for using the GNU compiler tools and the second one is the file **sclv2_VS.exe** for using the Visual Studio C development tools. Download the archive from the following web page.

`ksuweb.kennesaw.edu/~jgarrido/SCL_programs`

2. Extract all files from the archive.
3. The subdirectory **progs**, contains the sample SCL programs and have the **scl** extension.

3 Using a Terminal or Command Window

3.1 Using Linux

1. On Linux, open a Terminal window and change to the directory where the SCL software is stored (for example, **SCL_progs**. Type the command:

```
cd SCL_progs
```

2. Extract the files in the archive and copy the compiler/translator, and subdirectory with the SCL sample programs to this directory. Type the **ls** command, and this will list all files and subdirectories in the current directory.
3. To compile an SCL program in file **linked.scl**, change to the **progs** subdirectory (**cd progs**) and type the following command:

```
../sclv2.out linked.scl
```

This is equivalent to the command:

```
~/SCL_progs/sclv2.out linked.scl
```

4. Verify that the new file generated, **linked.c**, is in the current directory. Type **ls -lt**, and this will list all files and subdirectories in the current directory ordered by date and time.

5. Compile and link the generated file (`linked.c`) with the GNU C compiler tool set. To compile and link file `Carwash.cpp`, type the following command:

```
gcc linked.c -o linked.out -lm
```

For some programs, an external (C) library might be necessary to link to the program. For these programs, a simpler way to compile and link is using a shell file such as `compl.sh`; type the command:

```
./compl.sh linked.c
```

This will generate the executable file `linked.out`.

6. To run the executable file generated, type the command:

```
./linked.out
```

3.2 Using Windows and MinGW GNU Compiler Tools

On Windows, open a command window by typing `cmd` on the Search option. Check and verify that the path is complete. Type:

```
path
```

To verify access to the GNU C/C++ compiler tools, type:

```
gcc
```

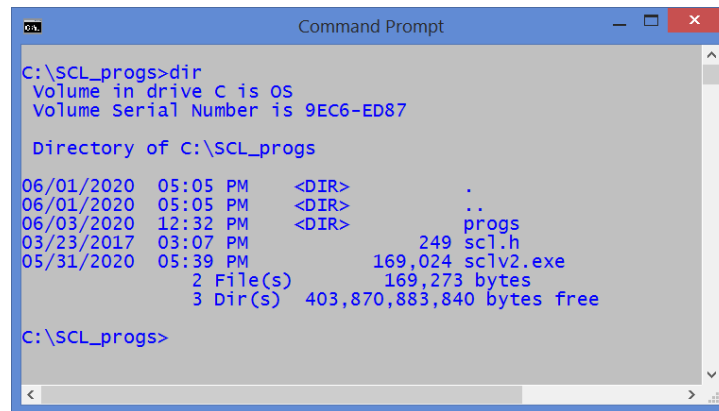
If the GNU C/C++ compiler tools are not accessible, then an extension of the path is needed. Type

```
gccpath
```

This run the batch file `gccpath.bat` and will expand the path by including the directory where is compiler is stored. You might need to edit this file. To verify access to SCL translator, type:

```
sclv2
```

If Windows replies that some 'dll' files are missing (`libgcc_s_dw2-1.dll` and `libstdc++-6.dll`), these need to be downloaded and installed on the `mingw` folder, under the `bin` subdirectory. Restart the system after that and should run with no problems. Figure 1 show a typical command window.



```
C:\SCL_progs>dir
Volume in drive C is OS
Volume Serial Number is 9EC6-ED87

Directory of C:\SCL_progs

06/01/2020  05:05 PM  <DIR>          .
06/01/2020  05:05 PM  <DIR>          ..
06/03/2020  12:32 PM  <DIR>          progs
03/23/2017  03:07 PM                249  scl.h
05/31/2020  05:39 PM            169,024  sclv2.exe
                2 File(s)            169,273 bytes
                3 Dir(s)  403,870,883,840 bytes free

C:\SCL_progs>
```

Figure 1: Command window.

1. On Windows, open a command window by typing `cmd` on the Search option. Change to the directory where the SCL software is stored. Type the command:
`cd SCL_progs`
2. Extract the files in the archive and copy the compiler/translator, and subdirectory with the SCL sample programs to this directory.
3. Verify that the compiler/translator, and the sample SCL programs are in the current directory. Type `dir`, and this will list all files and sub-directories in the current directory.
4. The sub-directory `progs` stores all the SCL programs. These have an `.scl` extension. Change to the sub-directory by typing `cd progs`, then type `dir` to verify that all programs from the downloaded archive are present.
5. To compile the program in file `linked.scl`, type the following command:

```
C:\SCL_progs\sclv2 linked.scl
```

6. Verify that the new file generated, `linked.c`, is in the current directory. Type `dir /p/o-d`, and this will list all files and sub-directories in the current directory ordered by date and time.
7. Compile and link the generated file (`linked.c`) with the GNU C compiler. To compile and link file `linked.c`, type the following command:

```
gcc linked.c -o linked.exe -lm
```

For some programs, an external (C) library might be necessary to link to the program. For these programs, a simpler way to compile and link is using a shell (batch) file such as `compl.bat`; type the command:

```
compl linked.c
```

This will generate the executable file `linked.exe`.

8. To run the executable file generated, type the command:

```
linked
```

3.3 Using Windows and Visual Studio C/C++

1. On Windows, open a VS command window by typing `cmd` on the Search option and select Developer Command Prompt for VS 2017.
2. Extract the files in the archive and copy the compiler/translator, and sub-directory with the SCL sample programs to this directory.
3. Change to the directory where the SCL software is stored, for example `SCL_progs`. Type the command:

```
cd SCL_progs
```

4. Verify that the subdirectory `progs` stores the SCL programs. These have an `.scl` extension. Change to the subdirectory by typing `cd progs`, then type `dir` to verify that all SCL programs from the downloaded archive are present.
5. To compile the SCL program in file `welcome.scl`, type the following command:

```
C:\SCL_progs\sclv2vs welcome.scl
```

6. Verify that the new file generated, `welcome.c`, is in the current directory. Type `dir /p/o-d`, and this will list all files and sub-directories in the current directory ordered by date and time.
7. Compile and link the generated file (`welcome.c`) with the VS C/C++ compiler tool. To compile and link file `welcome.c`, type the following command:

```
cl welcome.c
```

For some programs, an external (C) library might be necessary to link to the program. For these programs, a simpler way to compile and link is using a shell (batch) file such as `complvs.bat`; type the command:

```
complvs welcome.c
```

This generates the executable file (by default) `welcome.exe`.

8. To run the executable file generated, type the command:

```
welcome
```

4 Preparing CodeBlocks

The following sequence of steps involve the basic procedure for setting the appropriate options and configuring Codeblocks for using the SCL translator then the remaining steps for editing, compiling, and linking C programs.

4.1 Download the SCL Translator

The preliminary step required is the installation of the the SCL translator and the sample programs. The versions of SCL translator (for Linux and Windows) are available on the web page mentioned previously.

1. For Linux, download the archive file `SCL_progs.tar.gz` from the web page and extract all files from archive.
2. For Windows, download the archive file `SCL_progs.zip` from the web page and extract all files from archive.
3. Select the executable file of the SCL translator (`sclv2.out` for Linux or `sclv2.exe` for Windows)

This executable file of the SCL translator, the `scl` source files, and the script files must be stored on a folder such as `~/SCL_progs`.

4.2 Setting the Codeblocks Options

1. Start CodeBlocks, the first screen that appears. Click on ‘Create a new project’.

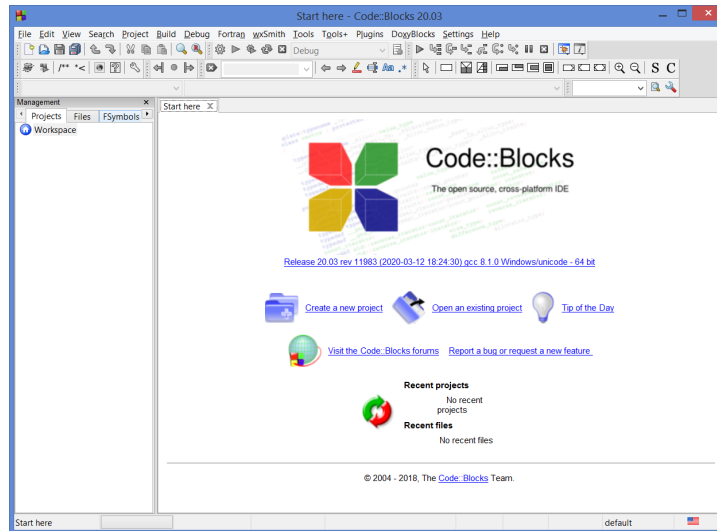


Figure 2: Starting Codeblocks.

2. On the top bar click on the Settings menu, then select Compiler. A dialog box appears. On the tab group, activate the Other settings tab, and click the button Advanced options located on the lower right of the dialog box. See Figure 3.

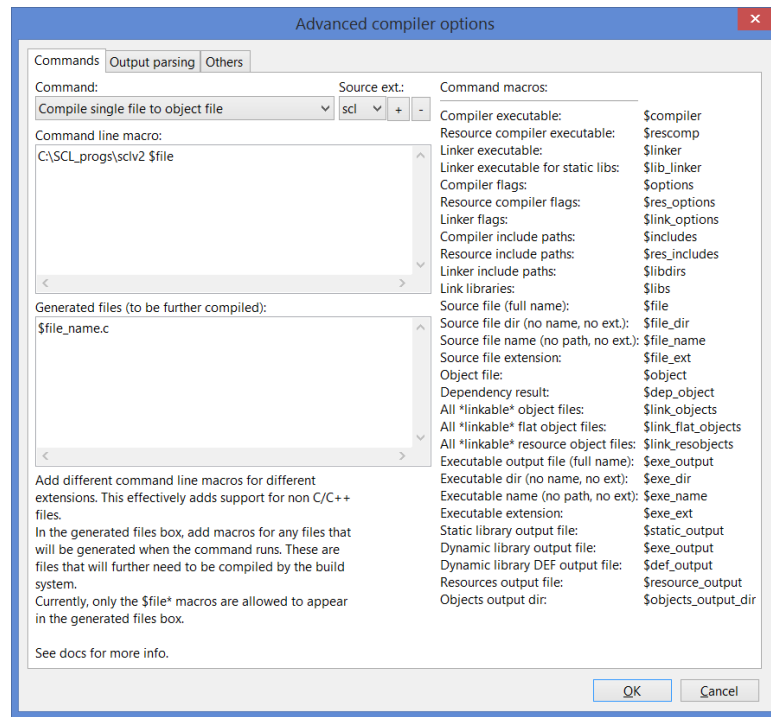


Figure 3: Selecting Advanced options.

3. Click the Yes button when the box *Edit advanced compiler setting?* appears.
4. On the new window, under Source ext: click the '+' button and type `scl` in the dialog box.
5. Type the Command line macro and the Generated files. The window will now appear as shown in Figure ?? . This assumes that the SCL translator (executable file `scl`) is located in folder: `~/SCL`
6. Click the Ok button located on the bottom of the window.

Codeblocks is now setup to recognize source files with an `scl` extension for editing and invoking the SCL translator.

4.3 Using Codeblocks for Implementing SCL Programs

1. Select Console Application and click the Go button, which is located on the upper right corner of the window.

2. On the Console Application window, click the Next button. Select C language and click Next.
3. Type the project title (name), e.g. *Welcome*. In this example, the project will be created in folder: `/home/jgarrido/comp_models/scl_models`, as shown in Figure 4. Click the Next button.

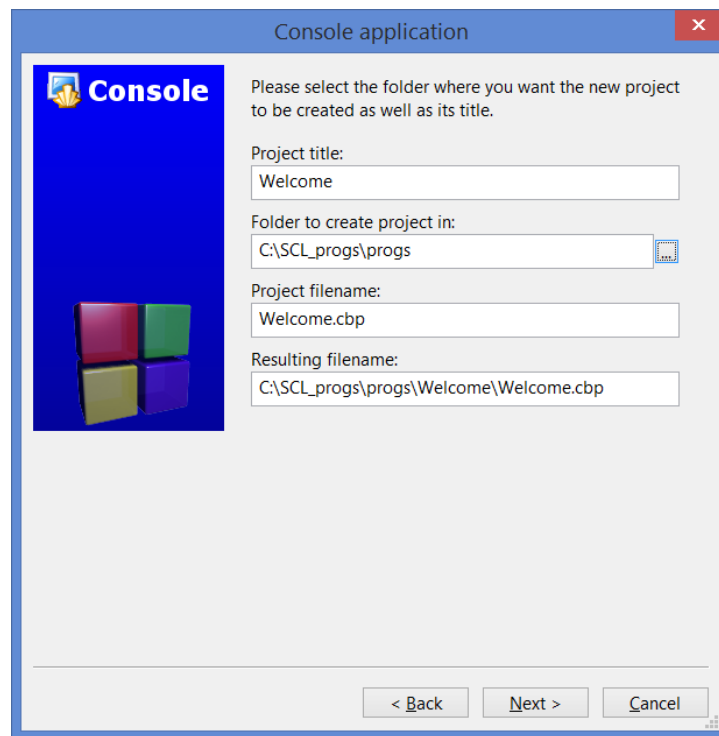


Figure 4: Title and location of project.

4. Select GNU GCC Compiler and click the Finish button.
5. Activate the left pane of the screen (Management), click the Projects tab. Remove the C source file 'main.c' by right-clicking on it.
6. A new source file can be created by selecting File menu, then New, and Empty file. Click Yes to add this empty to the project. A new dialog window appears, type the name of the file with its `scl` extension. Now you can start editing this source file and when finished, save the file.

7. If one or more existing source files are to be included in the project, select the Project menu on the top bar, or right-click on the project name. Select Add files. Select the directory of the source to add to the project and select the SCL source file(s). Click Open, see Figure 5.

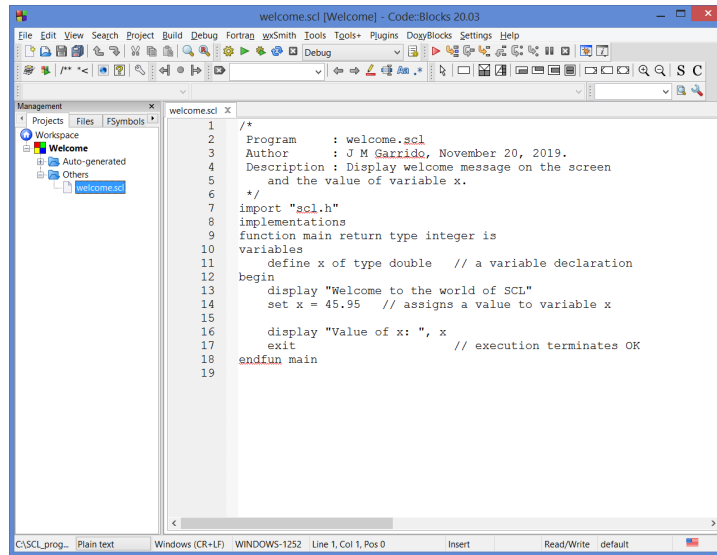


Figure 5: Editing an SCL source file.

8. The source files are now under 'Others' and under the project name. Double-click on the desired source file to edit it further. The file now appears on the edit area of the screen, as shown in Figure 6.

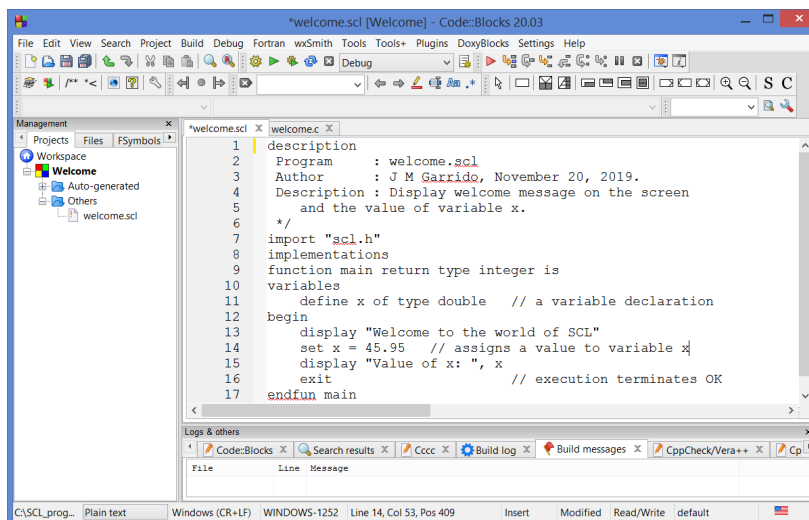


Figure 6: SCL programs.

9. Right-click on the current project name, or activate the Project menu in the top bar, and select 'Build options'.
 - (a) On the tab 'Compiler settings', click on the Compiler Flags tab and check 'Enable all common compiler warnings'.
 - (b) This is an optional step, for small programs may not be required. On the tab 'Linker settings', click the 'Add' button to add a library to the project. Repeat to add all necessary object files and libraries. For example, for a program `Plineq_sol3`, a local library `basic_lib`, and the `m` (standard math library). This is shown in Figure 7.

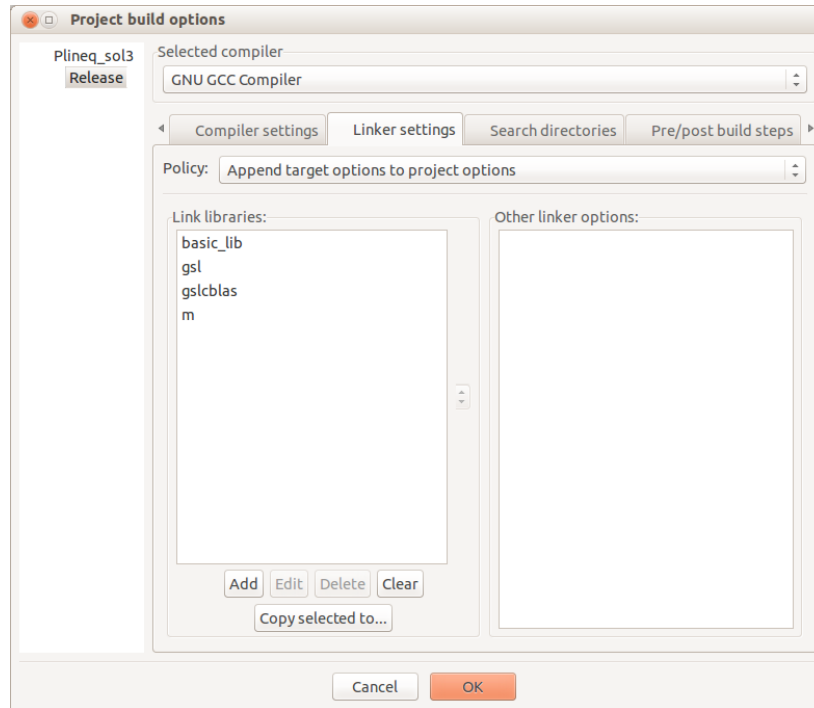


Figure 7: Linker settings.

- (c) On the tab 'Search directories' and the tab 'Compiler', add the search directory for header files that may be required by the source program while compiling.
 - (d) On the tab 'Search directories' and the tab 'Linker', add the search directory of the libraries if needed. On Linux Ubuntu, the external libraries are always stored in standard system directories. The local library '*libbasic-lib*' library can be stored in any location, for example `~/comp_models/basic`. For this, click the Add button, then click (...) and navigate to the appropriate directory.
10. Build the project, which translates the SCL source file, compiles, and links the corresponding C files in the project. On the top bar, select the Build menu and select the Build option. The Build log appears in the lower pane of the Codeblocks screen and is shown in Figure 8.

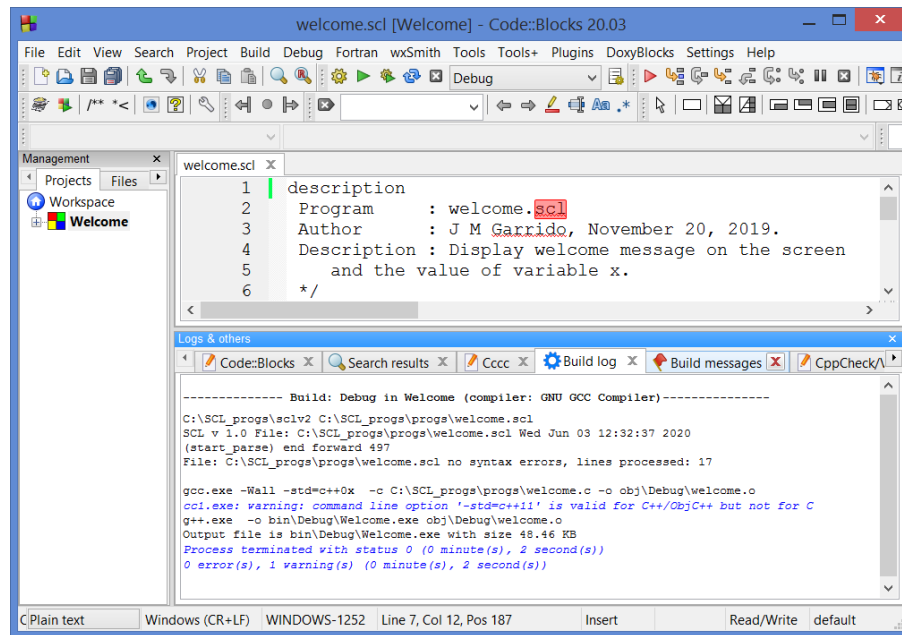


Figure 8: Building the project.

11. To execute the program, select the Run option in the Build menu. A new screen appears with the results of the execution, as shown in Figure 9. After the program terminates execution, press the Enter key.

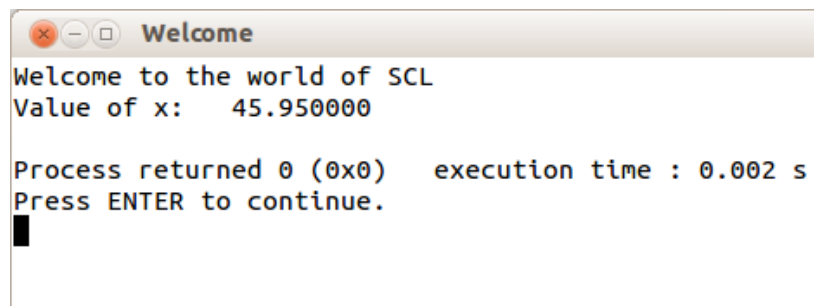


Figure 9: Executing the project.

12. On the top bar, activate the File menu and select Close project. Codeblocks creates several new directories and the executable file is located in the Windows directory C:\SCL_progs\progs\Welcome\bin\Debug or on Linux: ~/SCL_progs/progs/Welcome/bin/Debug.