Appendix A

Further information about using Code::Blocks

A.1 Introduction

This appendix gives you more detailed guidance on various important aspects of the Code::Blocks "clab" project that has been created for you to do C programming. There is further infomation about the various software libraries that we have integrated into the "clab" project to allow you to handle graphics, audio and midi. In addition further guidance is provided on an number of useful topics such as creating your own targets, associating source files with targets and importing source files into targets.

The information in this appendix will be more important as your experience of programming in C develops and in the construction of your assessed assignment.

A.2 Graphics, audio and music software libraries

The C course has a set of core exercises for all students, but also contains exercises that are degree specific. Students on music technology related courses have some additional programming exercises that relate to the manipulation of audio files and the production of computer generated music using MIDI. Students who are not on music technology related courses have additional programming exercises that are based around graphics. However, both groups of students are encouraged strongly to utilize aspects of both graphics and audio in their programming assignment.

In order to circumvent OS specific code for graphics, audio and music(MIDI) we use a number of cross platform libraries that provide these functionalities; **Allegro** for graphics, **libsndfile** for reading and writing audio files to disk, **portaudio** for real-time audio capabilities and **portmidi** for MIDI. All these software libraries are widely used.

These libraries provide the developer with one common programming interface, hiding any system specific code. Although you will not deal directly with these libraries it is good to know that all the exercises on this course and the programs that you will develop use them internally. We have created some wrapper functions that utilise the Allegro (graphics) and libsndfile, portaudio and portmidi libraries. These functions make using these libraries relatively easy.

We chose these libraries for graphics, audio and MIDI for three main reasons:

1. They are written in pure C and are well suited for this course.
2. They are cross-platform libraries meaning that you can write your code once and compile (the same code) in multiple architectures like for example Windows, Apple OS and Linux.
3. They are free software which means that we can use them for educational purposes without having to pay any license fee.

Four files have been created:

- `graphics_lib.h` which uses Allegro internally and provides graphics capabilities. This must be included in all program that use allegro graphics.
- `amio_lib.h` which uses libsndfile, portaudio and portmidi internally and provides audio and MIDI capabilities. This must be included in all program that use audio or midi functions.
- `graphics_lib_functions.c` which must be associated with all targets that use graphics. This file gives the implementations of the wrapper graphics functions.
- `amio_lib_functions.c` which must be associated with all targets that use midi. This file gives the implementations of the wrappers functions for all audio and midi functions.

In C files that end in .h need to be included via a special instructions which begins with the hash symbol. For example:

```
#include <graphics_lib.h>
#include <amio_lib.h>
```

We will discuss these types of instructions later on in the course.

A.3 Project Build Options

Right click on your project (clab) and select the "Build options..." menu item. The "Project build options" window will open. On the left of that window you see a tree structure containing all targets of your project. Clab is your main project and you will see many targets: lab1, lab2 and so on. Each target has its own options (Selected Compiler and the settings in each individual tab i.e. compiler settings, linker settings etc.) The settings that we are interested in are the following:

- Selected compiler: This should always be set to "GNU GCC Compiler".
- Compiler settings: These settings should be left untouched. Look into the "#defines" tab and you should see "ALLEGRO _STATICLINK". This is the only setting that we changed and ensures that the allegro software libraries are statically linked. This allows you to end up with a single executable file that contains all the binaries (compiled files) inside of it, so when you open your program's folder you see one executable file along with the rest of the files related to your game (images, sound etc.).
- Linker Settings: In this tab all third party libraries are specified. The linker tries to link against the libraries that are found in this page. The "Link libraries" panel has all allegro related libraries. The "Other linker options" has some additional libraries related to allegro that we need to link against plus the portmidi and winmm libraries that we need for MIDI functionality. Some targets that do not use these libraries will not have any linker settings.
- Search directories: In this tab we have set up the directories that contain the include files (header files) and the libraries. The compiler will look into these directories to find any files needed for compiling the source and linking the object file against any needed libraries. First look in the "Compiler" tab. There are five directories : "include\allegro","include\portmidi", "include\portaudio", "include\libsndfile"and "include" containing the graphics and audio header files and the wrapper function header files. Now click on the "Linker" tab. There is only one directory called "lib" which contains all necessary library files (.a and .dll files).

Now close the "Project build options" window.

A.4 Project/Targets Options

Right click on your project (clab) and select the "Properties..." menu item (at the bottom).
The "Project/targets options" window will open. This window has many tabs but the only one we are

interested in is the "Build targets" tab. Click on the "Build targets" tab. This is where we manage all our targets (programs). On the left you should see the "Build targets" panel which lists all the current targets that are included in the project and has some buttons on the right for basic operations; these will be explained in a later section. Now focus on the "Selected build target options" panel. As the title suggests the settings that you see in this panel correspond to the selected target. Let us now look at each setting separately.

- Platforms: This is set to "All". It instructs the compiler which OS is the target being built for. Leave this as it is.
- Type: This is set to "Console application". The kind of programs that we will write are all console applications so leave as is. The "Pause when execution ends" box keeps the command window open when the program exits. It is checked by default but you can uncheck it if you want. It does not affect the behaviour of the program at run time.
- Output filename: This sets the location and file name of the produced executable. In this case it is set to "bin\Debug\lab1.exe". This means that when we build the "Debug" target the executable will be placed inside the "bin\Debug" folder and will be given the name "lab1.exe". The two boxes below are checked by default. Leave them as is.
- Execution working dir: This sets the location of the executable when it is run through Code::Blocks. It is set up to be the directory where the compiled executable resides.
- Objects output dir: This sets the location of the object files. Object files are produced by the compilation process. Leave as is.

  Now look at the "Build target files" panel. This is where we specify which source files belong to which targets. When "lab1" target is selected you fille find that the "lab1.c" file is selected, this means that the target only requires this file. Click on the graphics1 target and you will see that the files "src\graphics1.c" and "src\graphics_lib_functions.c" are selected in the "Build target files" panel. The file "graphics1.c" is the source file (it contains the `main` function). The file `graphics_lib_functions.c` is a library file that has the source code relating to the graphics functions. It must be part of all projects which use graphics functions. "Build target files" are source files (not ".h" files) that belong to the graphics1 target. Each target will have one or more files assigned/associated to it and this is the place where we can do this. As we will see later on, every time we add a new file in the project we will be given the opportunity to select the target that the imported/created file belongs to.

Close the "Project/targets options" window.

A.5 Creating a Project from a Target

Later on in this course when you develop your own programs you might want to create a separate Code::Blocks project for them. In order to do so:

- Execute "Project→ Properties..." from the menu bar.
- Select the Build targets tab.
- Choose one of the targets in the build targets panel
- Click on "Create project from target" button at the bottom of the build targets panel.
- Give your own name for your this project.
- Press ok.

Now using windows explorer navigate to your clab folder and you should see the Code::Blocks project you just created. If you double click on it while Code::Blocks is open (with the clab project already open) your new project will open on the same workspace. You can close any open project in the workspace by right clicking on the project and clicking on "close project". Also note that the new project will have the same settings as the target it originated from. You might like to verify that.

A.6 Creating new targets

1. Go to the build targets tab in the project/targets options window (right click on clab project and select properties).
2. Select the graphics1 target.
3. Press the "Duplicate" button.
4. Enter an appropriate target name (i.e graphics1_new).
5. Make sure that your newly created target is selected in the build targets panel. Change the output filename from graphics1.exe to your target's name (e.g graphics1_new.exe).
6. Close the project/targets options window. At this point you should have a new target called graphics1_new.

The targets: graphics1.c, graphics2.c, graphics3.c, music1.c, music2.c, and music3.c have the same settings (i.e. both allegro and audio/midi libraries are linked in). Some of the other targets do not have links to these libraries as the exercises are "pure C" exercises and do not require these libraries. So if you wish to create your own target and you know it will use graphics libraries, then it would be wise to copy and edit one of the graphics targets. Likewise if it is a midi program you wish to work on then it is best to copy and edit a music target. If the program uses neither graphics or midi libraries then copy and edit one of the targets that does not use them (typically "labX.c", where X is a lab number).

A.7 Associating Source Files to Targets and Creating a New Source File

After you create your targets you need to either import a source file or create a new one. When doing that you have to associate the file with a target.

- Execute "File→ New → Empty file" (Ctrl+Shift+N).
- Press "yes" to "Add file to project".
- Give the source file an appropriate name (e.g. myprog.c) and save it inside the src directory of your clab folder. This way all your source files are organised. You can create any additional folders inside the src folder, if you wish; anything that suits you.
- The multiple selection message will pop up. This is the step where we can select the target that this file should correspond to. Only select the targets you created in the previous section. (graphics _new in my example) and deselect everything else.
- Press "ok".
- Your file should appear in the projects tab in the management panel and also open in the editor panel.

At this point you are ready to start writing code and building applications.

A.8 Importing a Source File

1. Right click on the clab project (in the projects tab, management panel).
2. Click on "Add files"
3. Select the required source file. It is advisable to have copied your source file inside the src folder of your clab project (using windows explorer). This ensures that when copying/moving your clab project around all your files are copied/moved together.

4. The multiple selection message will pop up. This is the step where we can select the target that this file should correspond to. Only select the targets you created in section A.6
5. Press ok.
6. Your file should appear in the projects tab in the management panel and also open in the editor panel.

At this point you can test your imported source file by building and running the targets.