Tutorial: Create 2D Game Engine using C++

URL: <https://www.youtube.com/playlist?list=PL-K0viiuJ2RctP5nlJlqmHGeh66-GOZR_>

# Purpose

These notes summarize the you tube video playlist “How to Make a 2D Game Engine with C++ & SDL”. We will try to follow the videos in order but will not hesitate to re-arrange the material to make it easier to read and follow and understand the concepts presented.

In addition, we will use the latest version of all software and libraries and note the changes.

# 1. Setup SDL, SDL\_image, SDL\_ttf in Code::Blocks

The first video walks you through obtaining and installing Code::Blocks. In addition, it creates the initial project for this series, which of course starts as a “Hello, World” project. Lastly, it sets us up to use the SDL libraries.

## Why use Code::Blocks?

Code::Blocks is a free open-source, cross-platform Integrated Development Environment (IDE) primarily designed for C, C++ and Fortran programming. The IDE was originally released in 2005, and its development is driven by a team of contributors using wxWidgets library for its graphical user interface (GUI). The first official stable version was 8.02 released in 2008. The release marks the year and month of the Code::Blocks version. Like most modern IDEs it provides for extensions via a plugin system.

## Install Code::Blocks

The official website to obtain Code::Blocks is <https://www.codeblocks.org/downloads/> . You have the option of:

* Downloading the setup file for Code::Blocks and executing it.
* Download the source code and build from scratch

The Binary release has a Windows, Linux and Mac OS X version. The fact that there is a version of Code::Blocks that runs on all three platforms is the primary reason it has been selected for this tutorial. At the time of this writing the latest version is 25.03, which means it was released in March of 2025.

My choices for Windows are:



Figure - Selecting the right setup.exe to download

I decided to download codeblock-25.03mingw-setup.exe for 64-bit. I am surprised that the project has not moved to GitHub.

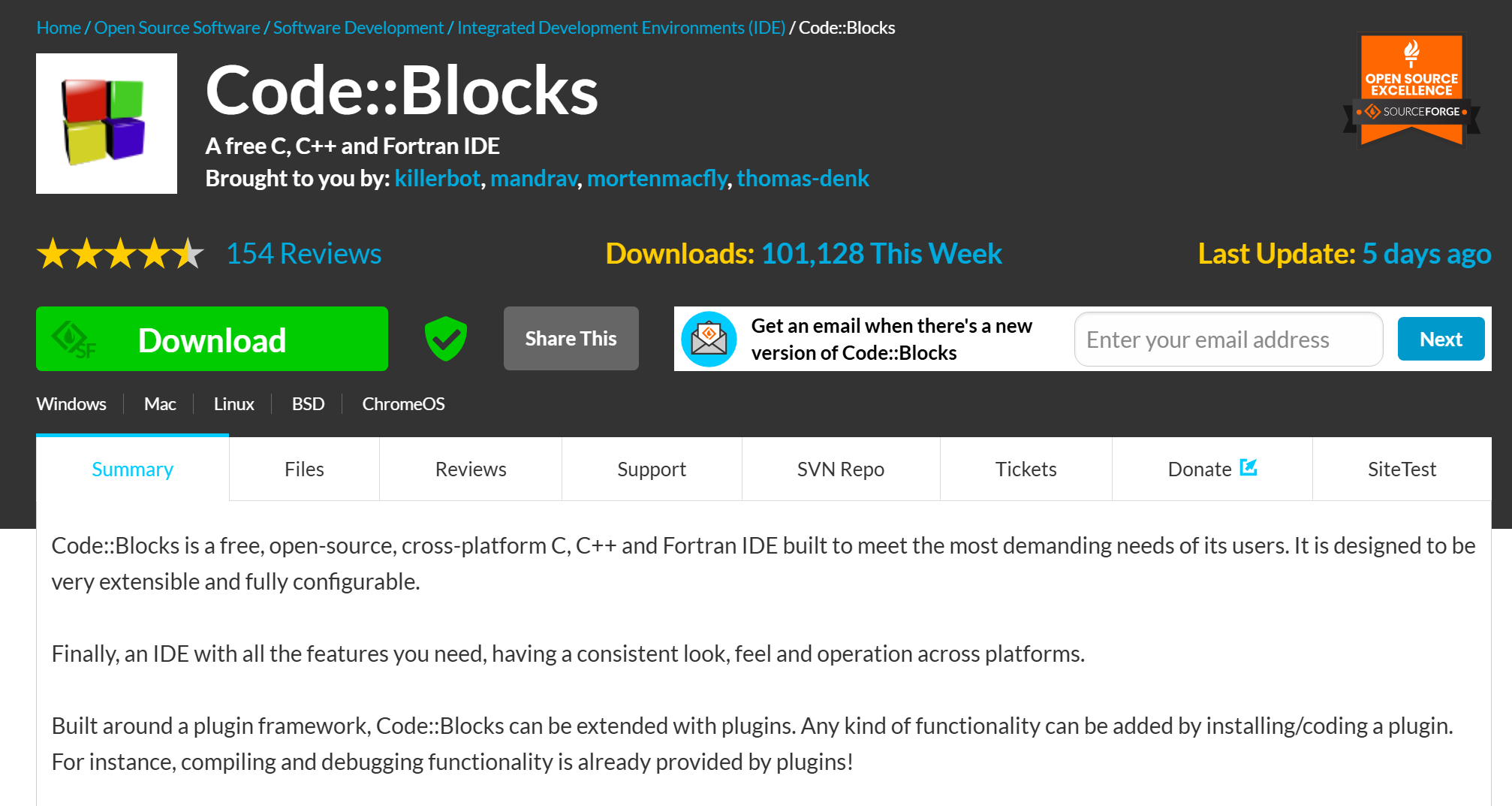


Figure - Downloading file from Sourceforge

I then run the setup.exe file:



Figure - Locating and executing the setup.exe file

### Running the Setup

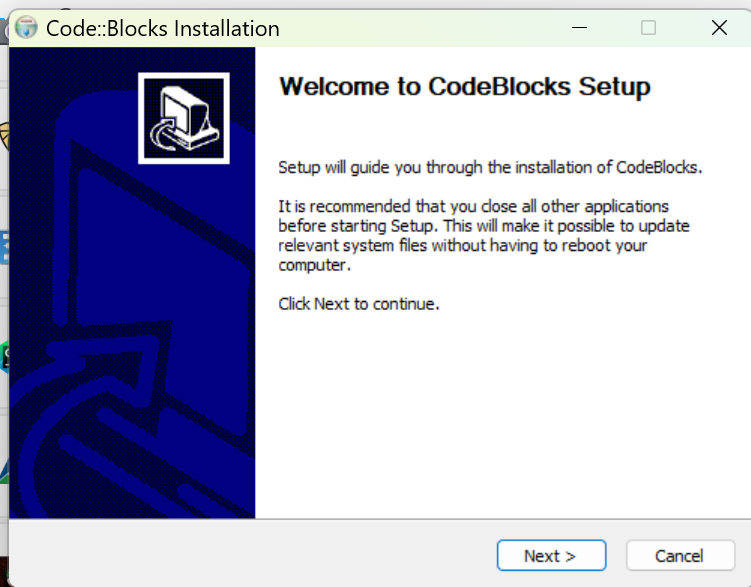


Figure - Initial Code::Blocks Installation dialog

I did not find it necessary to close any other applications. I just pressed “Next >”.

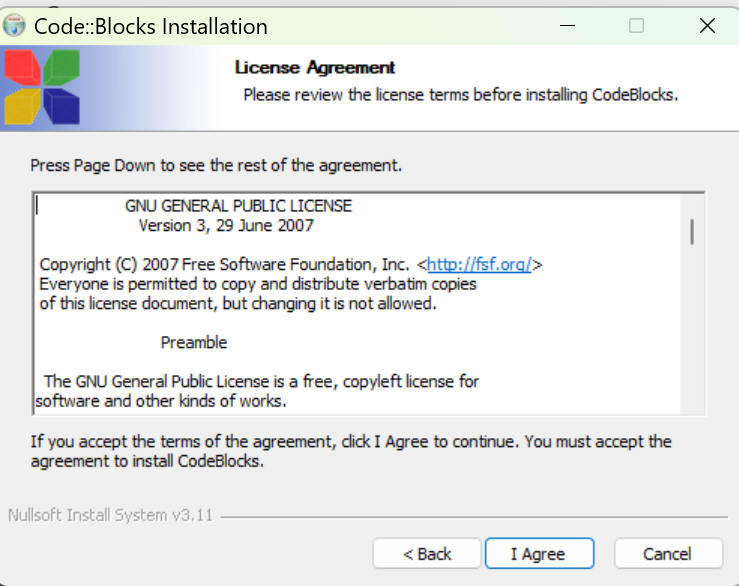


Figure - The License Agreement screen

OK, I did not read the license agreement but knowing it is the GNU license assures me that it is open source and free. I just clicked on “I Agree”.

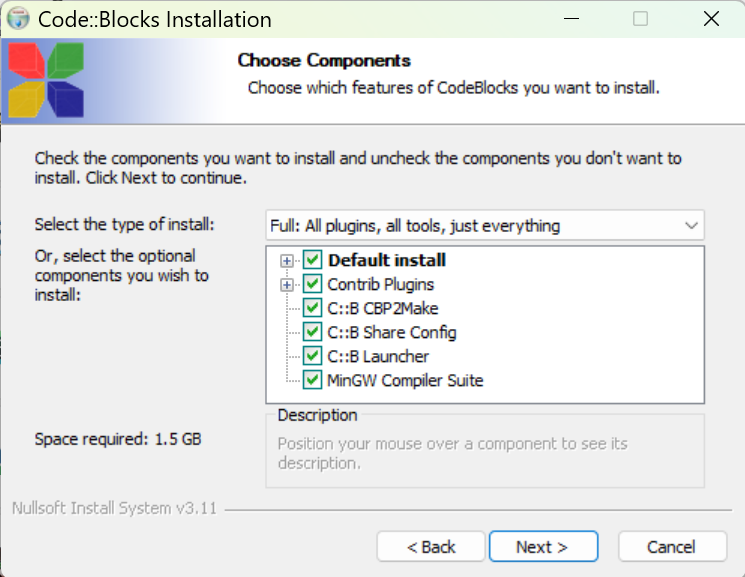


Figure - The "Choose Components" screen

It appears by default all the components are selected. A quick description of each component:

* **Contrib Plugins** – these are additional plugins developed by the community to extend the functionality of the Code::Blocks IDE. These plugins are not part of the core set but have proven to be valuable enough to be included in the official Code::Blocks repository.
  + **Code Snippets Plugin**: Helps to manage and insert reusable code snippets.
  + **DoxyBlocks Plugin**: Integrates Doxygen for generating documentation from your code
  + **CppCheck Plugin**: Provides static code analysis to identify potential bugs or issues
  + **SpellChecker Plugin**: Checks spelling in comments and string literals
  + **Valgrind Plugin**: Integrates Valgrind for memory debugging and profiling
* **C::B CBP2Make** – is a tool designed to generate Makefiles from Code::Blocks project files (\*.cbp) or workspace files. Essentially, it allows you to convert your Code::Blocks projects into Makefiles that can be used with GNU Make or other build systems.
* **C::B Share Config** – this is a tool that allows you to import and export parts of your Code::Blocks configuration. It’s particularly useful when you want to transfer settings between different computers or configurations.
* **C::B Launcher** – is a utility that helps manage the launching of the Code::Blocks IDE. This tool is useful for advanced users.
* **MinGW Compiler Suite** – The MinGW (Minimalist GNU for Windows) is a development environment that provides a native Windows port of the GNU Compiler Collection (GCC). It allows you to build native Window applications without relying on third-party runtime libraries.

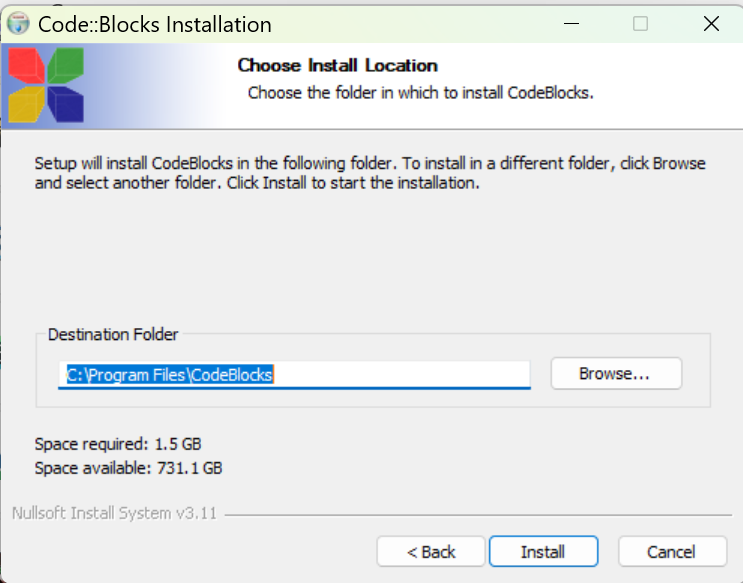


Figure - Installation location on your PC

I usually take the default location.

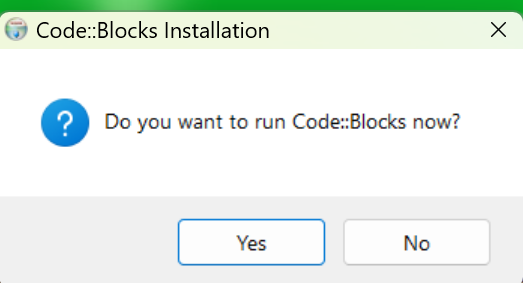


Figure - Prompt to run Code::Blocks now?

I clicked on “Yes” so I can enter a simple “hello world” program to make sure everything works.

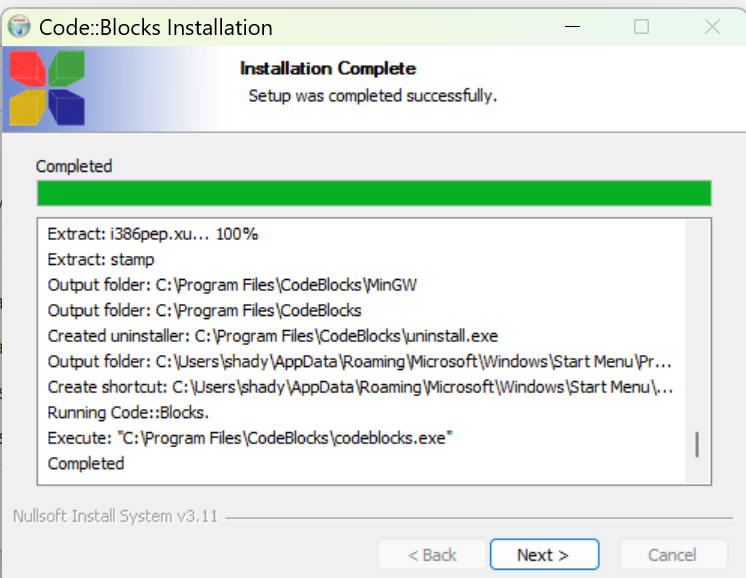


Figure - Code::Blocks installation completed dialog

I clicked on “Next >”.

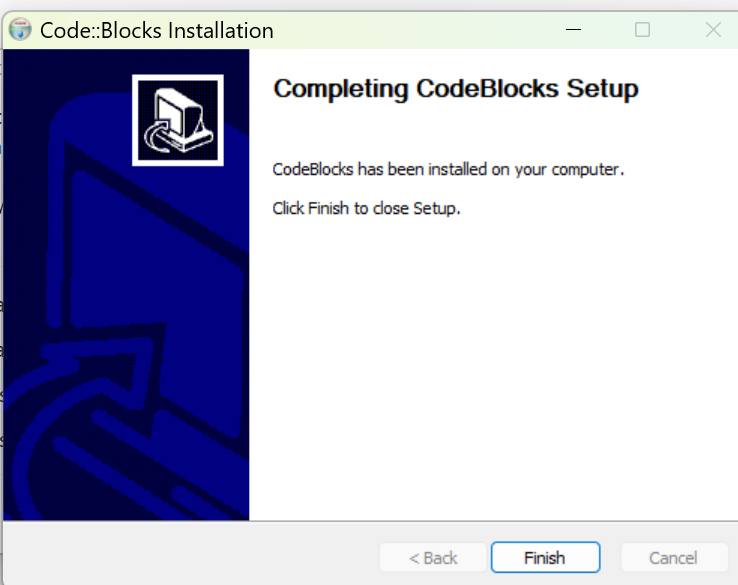


Figure - The final Code::Blocks setup screen

Click on “Finish” and you will see Code::Blocks starting if you clicked “Yes” to start Code::Blocks now.

### Starting up Code::Blocks

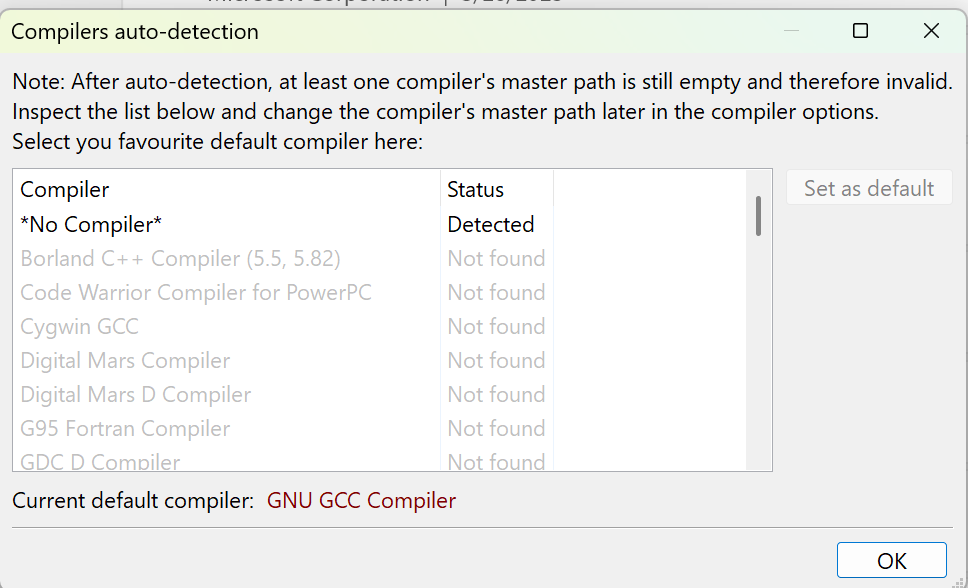


Figure - Code::Blocks lists all the compilers it detected

As you can see from the list you have the option of using many different compilers in Code::Blocks. We do expect Code::Blocks to find MinGW compiler because that is the version we downloaded.

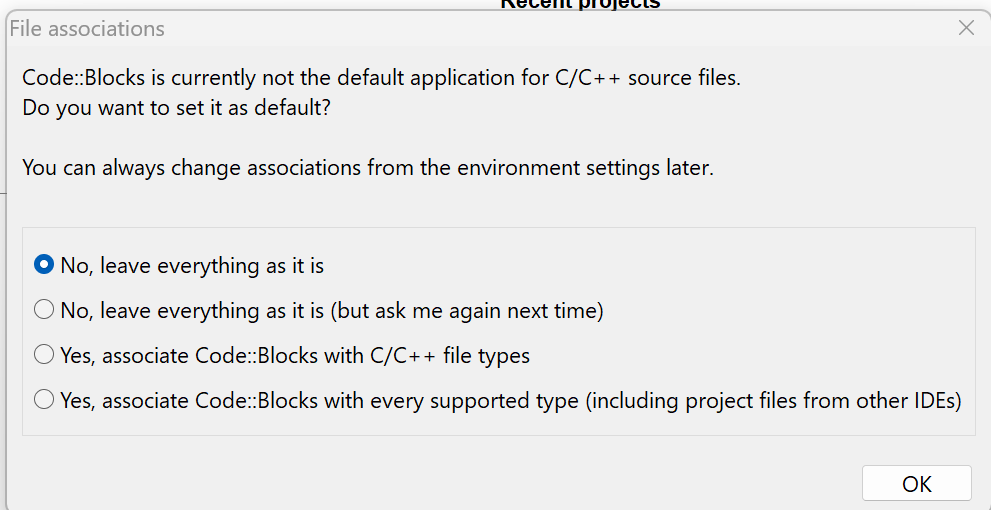


Figure - Option to update file association

I opted for the default of “No, leave everything as it is” since I do use Visual Studio 2022 for other projects.

## Starting a Project with Code::Blocks

A screenshot of a computer

AI-generated content may be incorrect.

Figure - The starting Code::Blocks screen

Note: I downloaded the Code::Blocks manual but the screenshots of the application appear to be dated!

### Making Adjustments to the screen

Another note: The screen icons and fonts appear to be small to me (it is probably due to my screen resolution). I made things larger by doing the following:

* Go to Settings 🡺 Environment 🡺 View
* Increase Message logs’ font size the Toolbar icon size

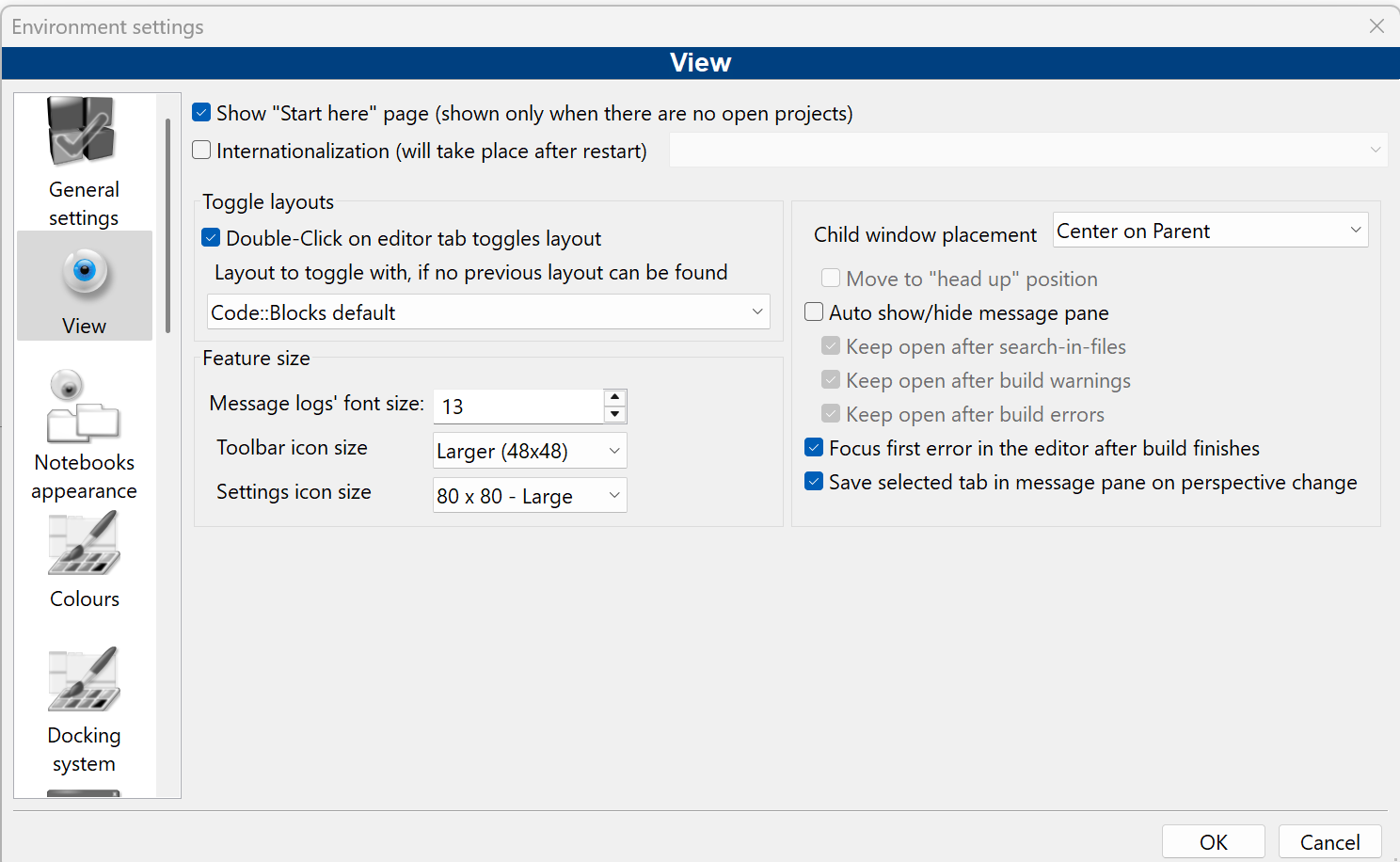
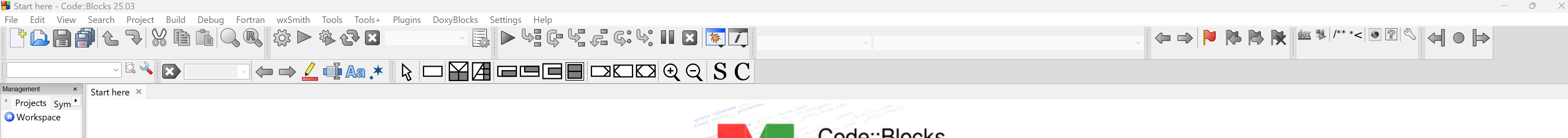


Figure - Increasing the toolbar icon sizes

* Click “OK”
* Select File 🡺 Save Everything
* Restart Code::Blocks



I also increased the font size used by:

* Select Settings 🡺 Editor

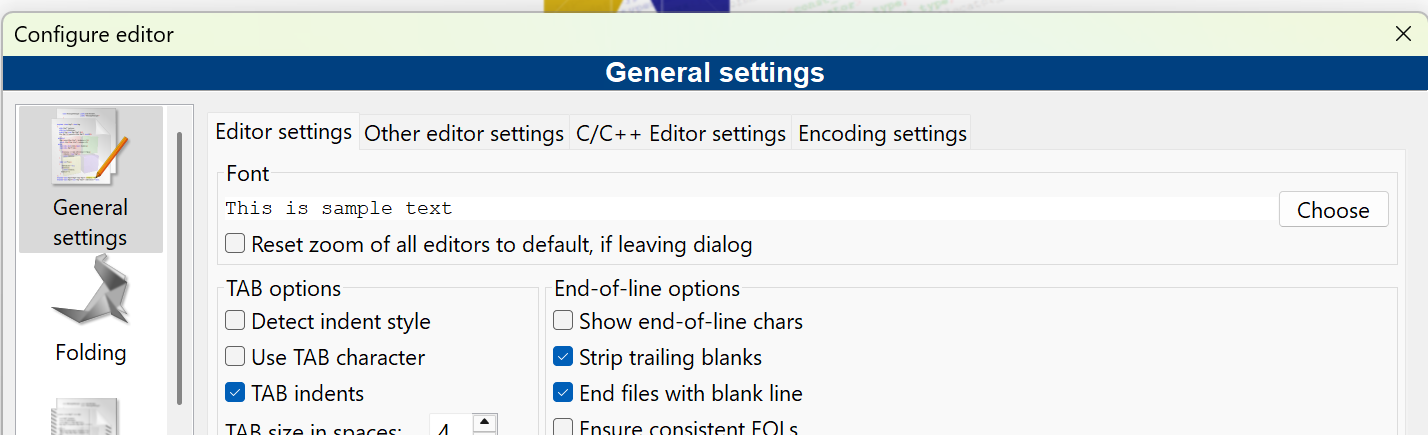


Figure - Updating font-size

* Click on “Choose” button

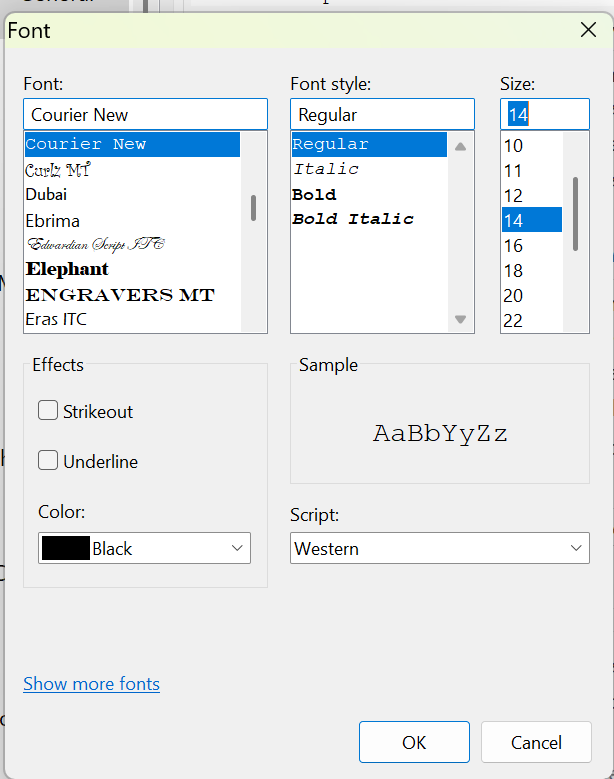


Figure - Select desired font size

* I selected 14 and clicked on “OK”

### Creating the Project

* Click on “Create a new project” link



Figure - Creating a new Project

* Select “Console Application”

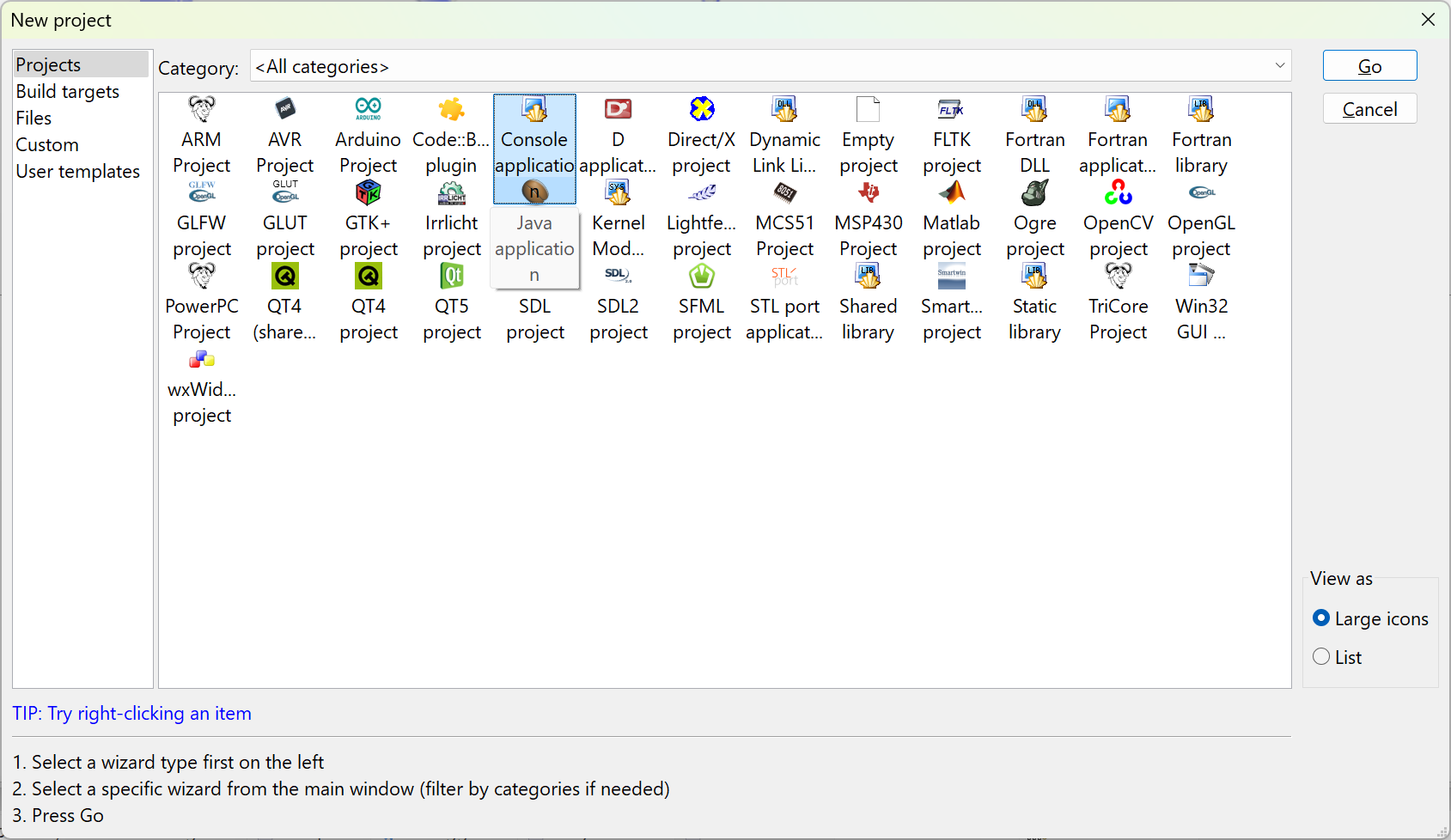


Figure - Creating our first console application

* Click on “Go”, if this is your first time you will see the following dialog box appear:

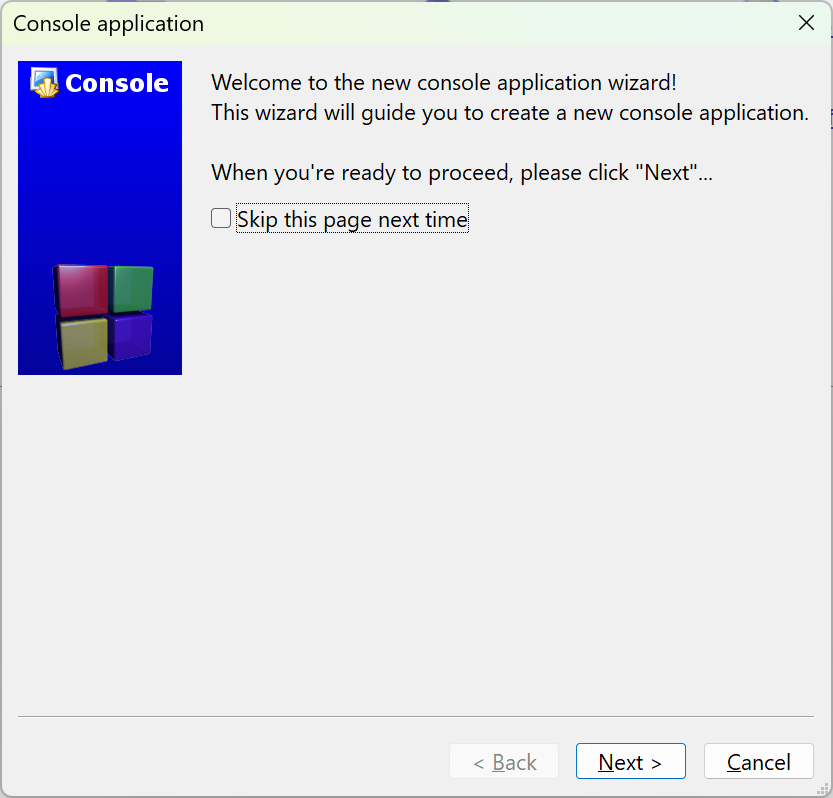
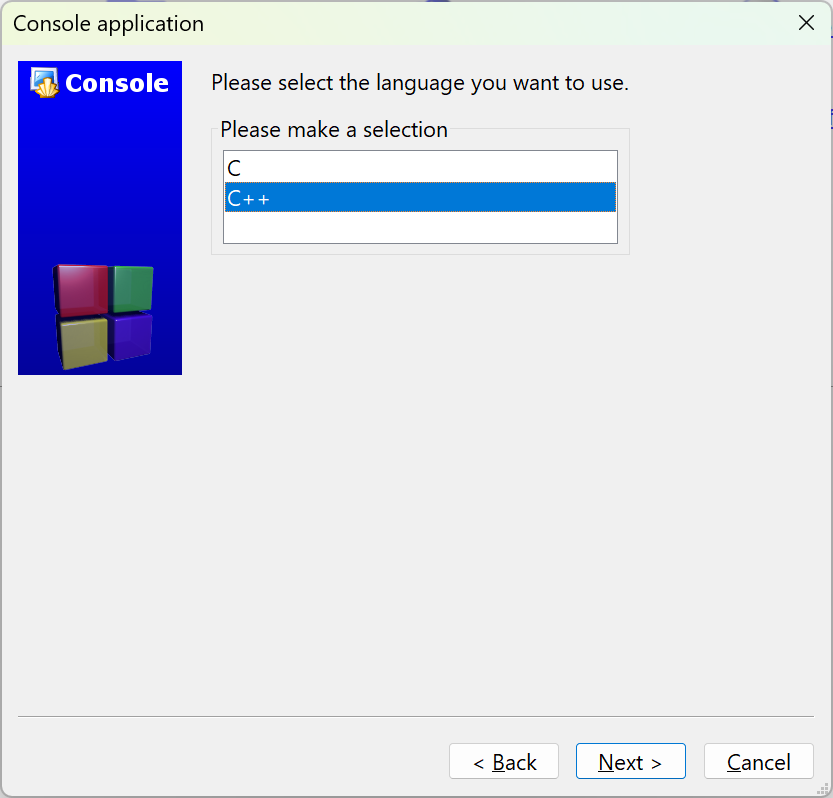


Figure - The Console application wizard starting...

* Click on “Skip this page next time” and press on “Next >”



* Take the default C++ and click on “Next >”
* Fill in the Project information:

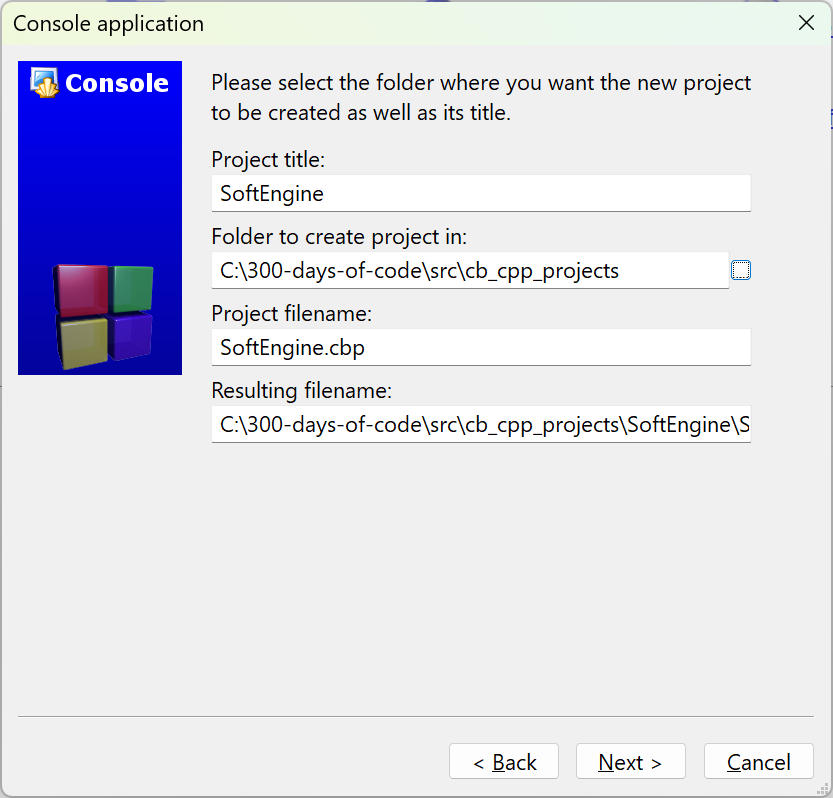


Figure - Project information

The Project name will be SoftEngine. Since this is part of my 300-days-of-code effort I select a folder where I plan on placing all my Code::Blocks C++ projects. Enter a location that makes sense for your setup.

* Click on “Next >”

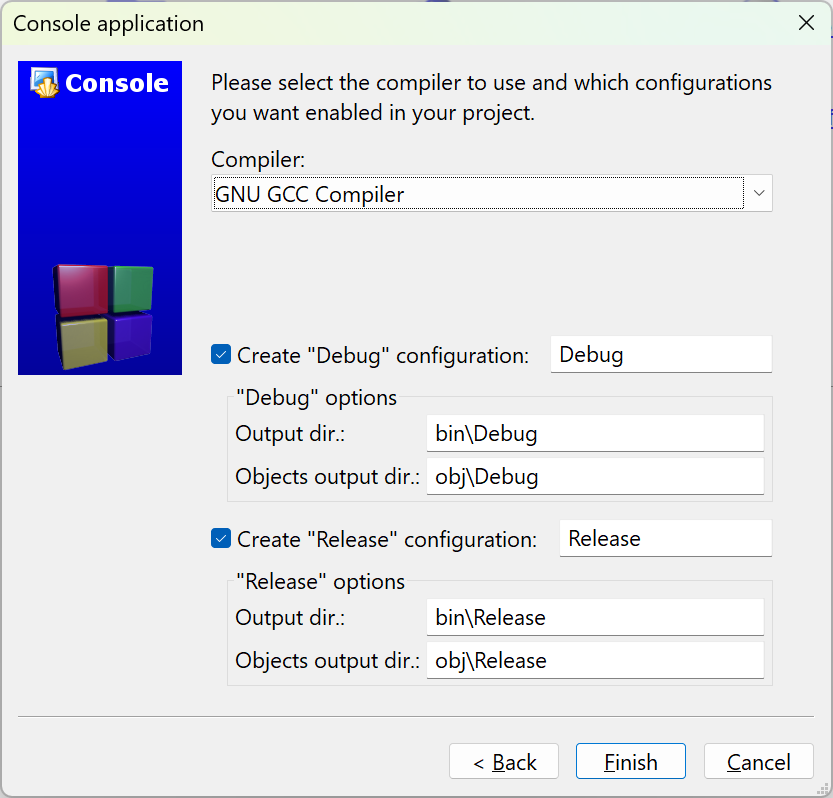


Figure - Final step in creating our game engine project

* Make sure the “Debug” and “Release” configuration are selected and click on “Finish”

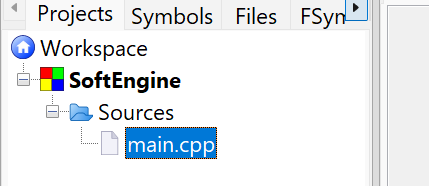


Figure - Default main.cpp is created

* Following the video author’s convention, let’s rename the file main.cpp to Main.cpp
  + Right-click on the filename and select Rename file:

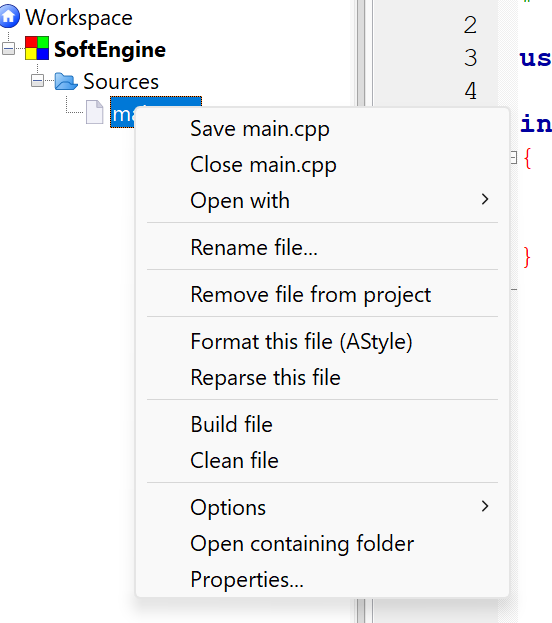


Figure - Context menu for file

* Enter the name Main.cpp:

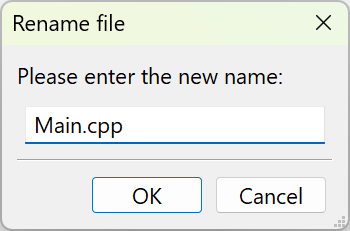


Figure - Rename dialog box