Code::Blocks Student Manual

Lawrence Goetz, Network Administrator

Yedidyah Langsam, Professor

and

Theodore Raphan, Distinguished Professor

Dept. of Computer and Information Science Brooklyn College of CUNY © 2006

Introduction

Through the aid of a **compiler**, a program written in a computer language, such as C++, is turned into machine code, which is executed on the computer. However, going from an idea to a program that works successfully in many instances takes a lot of time and effort. It may take several rewrites of code to get the program to work correctly. To accomplish this, students must learn a disciplined approach to organizing the code and learn how to trace their programs. The purpose of this manual is to help the student develop the skills to organize program coding and develop sound techniques for finding and isolating errors. Here you will learn how to trace the code step by step, so that it becomes clear where the problem is and why your program does not execute properly. This is called **debugging** the program. Hand tracing is useful in helping beginners understand where the bugs are and correct the program appropriately. Automatic tools have also been developed to help you trace programs that you have written and will be an important tool as your programs become more complex. This type of tool is called a debugger. A debugger lets you pause a program, while it is in the middle of running, and watch what is going on. Some debuggers work as command-line line debuggers, but newer debuggers have a nice graphical user interface, which is useful in helping you watch variables that you have defined as the program executes. The graphically-based debugger environment is part of what is called the Integrated **Development Environment (IDE)**. The purpose of these notes is to introduce you to this environment and help you learn how to use it as you develop and hone your programming skills.

A debugger cannot solve your problems for you. It is merely a tool to assist you when programming. You should first attempt to read over your code and using paper and pencil analyze the code to get an understanding of what is going on. Once you have gotten an idea of where in your code you have an error, you can then set the debugger to **watch** certain variables in your program. Watching your code will show you step by step how your program is being executed.

The debugger that you will use is part of an Open Source free IDE called **Code::Blocks**, which we have found easy to use and is described in these notes. Code::Blocks has a C++ editor and compiler. It will allow you to create and test your programs from one easy to use application. We hope these notes will assist you in making programming more enjoyable and help you develop better programming skills.

Table of Contents

Introduction	2
Installing the compiler	5
Installation of Code Blocks	
Starting Code::Blocks	
First Project	
Adding Files To Your Project	
Debugging a Program	

The first step in using the Code::Blocks IDE is to download it. The IDE can be obtained either for Windows-based or Linux-based systems at:

http://www.codeblocks.org/

http://www.codeblocks.org/downloads.shtml

The Linux based IDE can be downloaded as an **.rpm** file (Red Hat Package Manager and can be installed using the rpm utility within Linux). You can download the open source, cross platform Free C++ IDE at the following locations.

For convenience, the Windows-based system has been configured for you as a self extracting executable installation file (.exe file) and placed at:

http://www.sci.brooklyn.cuny.edu/~goetz/codeblocks/

This site has two installation files. One file is called **MinGW**, which is the **C++ compiler** and another file **Code Blocks Nightly**, which can be downloaded to any folder, including the Desktop.

These two files will then be used to install the IDE.

The installation instructions explained on the following pages, refer *only* to the files downloaded from the self installing exe files found on <u>http://www.sci.brooklyn.cuny.edu/~goetz/codeblocks/</u>.

Installation

Installing the compiler

After downloading the two files, MinGW and The Nightly builds, the first step will be to install the MinGW compiler. This is accomplished by opening (clicking on) the install file (MinGW), bringing up the following window:



Choose Next to continue

📥 MinGW Install Program		_ 🗆 🗙
Directory Choose an installation folder and click Next to continue.		•
MinGW's files will be installed in the following directory:		
C:\Program Files\MinGW		
Disk space needed :	81 Mb	
Available disk space :	83 Mb	
Click 'Next' to continue.		
< <u>B</u> ack	Next >	E <u>×</u> it

You may select a different installation directory. However it's easiest to take the default. Choose Next to continue.

MinGW Ins	tall Program 🔀
?	The destination directory doesn't exist. Do you want it to be created?
	<u>Y</u> es <u>N</u> o

Choose Yes.

📥 MinGW Install Program	- IX
Confirmation You are now ready to install MinGW.	•
This program will install MinGW into C:\Program Files\MinGW.	
Click 'Start' to install MinGVV.	
< <u>B</u> ack	E⊻it

When you are ready, choose Start to start the installation. Please wait while the installation is preformed.

📥 MinGW Install Program		
End Installation completed.		
MinGW has been successfully installed.		
	< Back	E <u>x</u> it

Choose Next to go to the exit screen.

📥 MinGW Install Program	_ 🗆 🗙
Clickteam Install Creator Copyright © 1999-2003 Clickteam http://www.clickteam.com	
This install program has been created with a freeware version of Clickteam Install Creat simple and powerful install maker.	or,a
This screen does not appear in install programs created with a registered version. You register online on the <u>Clickteam Web site</u> .	can
Thank you for using Install Creator!	
< Back Next >	E <u>x</u> it

Choose Exit to exit the application.

Installation of Code Blocks

Next, open (click on) CodeBlocks install file and the CodeBlocks Nightly will begin installing as follows:



Choose Next to continue.

📥 CodeBlocks Nightly Install Program		
Directory Choose an installation folder and click Next to continue.		6
CodeBlocks Nightly's files will be installed in the following	directory:	
C¦VProgram Files\CodeBlocks Nightly		
Disk space needed :	32 Mb	
Available disk space :	83 Mb	
Click 'Next' to continue.		
< <u>B</u> ack	<u>N</u> ext >	E <u>x</u> it

You may select a different installation directory. However it's easiest to take the default. Choose Next to continue.

MinGW Ir	nstall Program 🔀
?	The destination directory doesn't exist. Do you want it to be created?
	<u>Y</u> es <u>N</u> o

Choose Yes.

📥 CodeBlocks Nightly Install Program	
Confirmation You are now ready to install CodeBlocks Nightly.	O
This program will install CodeBlocks Nightly into c:\Progr	am Files\CodeBlocks Nightly.
Click 'Start' to install CodeBlocks Nightly. <u>= B</u> ack	<u>Start</u> <u>Exit</u>

When you are ready, choose Start to start the installation. Please wait while the installation is preformed.

📥 CodeBlocks Nightly Install Program	_ 🗆 X
End Installation completed.	•
CodeBlocks Nightly has been successfully installed.	
< Back	Exit

Choose Next to go to the exit screen.

📥 CodeBlocks Nightly Install Program			
Clickteam Install Creator Copyright © 1999-2003 Clickteam <u>http://www.clickteam.com</u>			
This install program has been created with a freeware version of Clickteam Install Creator, a simple and powerful install maker.			
This screen does not appear in install programs created with a registered version. You can register online on the <u>Clickteam Web site</u> .			
Thank you for using Install Creator!			
< Back Next >	E <u>x</u> it		

Choose Exit to exit the application.

When you have finished the Compiler and CodeBlocks installation, you should have two folders in the **Programs Files** folder: MinGW and CodeBlocks Nightly.

After installing Code::Blocks, and icon will be generated on your desktop, which is a shortcut to the Code::Blocks program:



Starting Code::Blocks

After you open up Code Blocks, you should setup the association of C++ to be used within the Code Blocks' IDE. However if you are using another C++ IDE and do not wish to associate C++ files with Code Blocks, you may skip the following steps.



From the Settings menu, select Environment.

Environment settings		×
	General settings	
	 Show splash screen on start-up Allow only one running instance (needs application restart to take effect) Check & set file associations (Windows only) Set now Manage Check for externally modified files 	
General settings	Ignore Invalid Targets	
Note app	On application start-up On project load Open default workspace Open all project files Open blank workspace Open last open files Do not open any files	
	Shell to run commands in: //bin/sh -c	1
Figure an unity Antipy of the second	Terminal to launch console programs: xterm -T \$TITLE -e	
	OK Cancel	

To have Code Blocks associated with C++ files, click on the Set Now button. Then click on OK.

You can also make changes to the Code::Blocks editor by clicking on the **Settings** pull-down menu and clicking on **Editor**. However, unless your professor tells you otherwise, you should leave these settings alone.

First Project

After you have finished downloading and setting up the Code::Blocks system, you can be in to write code. Code::Blocks creates what is called a **Workspace** to keep track of the **project** you are working on. It is possible for you to be working on multiple projects within your workspace. A **project** is a collection of one or more **source** (as well as **header**) files. **Source** files are the files that contain the source code for your program. If you are developing a C++ program, you are writing C++ source code (.cpp files). **Header** files are used when you are creating **library** files (.h files). A **library** is a collection of **functions** that are called to perform specific tasks, such as doing math, etc.

Setting up a **project** allows you to keep track of all the files in an organized way. When first starting out in computer programming, generally your projects will consist of a single source file. However as you gain experience and work on more complex projects, you will have projects containing many source files and dealing with header files as well.

Start here - Code::Blocks v1.0 - 🗆 × wxSmith <u>T</u>ools File Edit View Search Project Build <u>D</u>ebug Plugins Settings Help Ctrl-Shift-N New Empty file Project Ctrl-O 🚞 Open Build target Open default workspace File Recent projects Custom Recent files From user template $\triangleleft \triangleright \mathbf{X}$ Import project ٠ 🔚 Save 🛃 Save as... Code::Blocks Ctrl-Shift-S Save all files Save project The open source, cross-platform IDE Save project as user-template × Save all projects le::Blocks 🚺 Code::Blocks Debug 4 ► Save workspace StylePlugin Autosave BYOGames CBProfiler CB_Koders * Save workspace as... rd CodeCompletion CodeSnippets CodeStatistics Compiler Close workspace s Debugger FilesExtensionHandler DevPakUpdater DragScroll ource Exporter HelpPlugin cbKeyBinder ScriptedWizard 🔝 ⊆lose file Ctrl-W vxSmith wxSmithMime wxSmithWizard WindowsXPLookNFeel Close all files Ctrl-Shift-W tartup script • Close project

To create a project, click on the File pull-down menu, open New and then Project.

This will bring up the **New from template** window. Opening (clicking on) **Console Application** will then allow you to write a program on the console. The other application are for developing more advanced types of applications. After selecting **Console application**, click on the Go button to begin using the Console Application Wizard.



Console application	×
Console	Welcome to the new console application wizard! This wizard will guide you to create a new console application When you 're ready to proceed, please click "Next"
	< <u>B</u> ack <u>N</u> ext > <u>C</u> ancel

Press Next to go to the next step.

Console application		×
🐻 Console	Please select the folder where you want the new project to be created as well as its title.	
	Project title: First Program	
	Folder to create project in:	
	Project filename: First Program	
	Resulting filename: <invalid path=""></invalid>	
	< <u>B</u> ack <u>N</u> ext > <u>C</u> ancel	

Start by filling in the Project Title. You will notice that the Project Filename automatically becomes the same name. If you wish, you can change the filename, but for simplicity leave it as is. To specify the location of the folder to contain the project, click on the "..." button (selected in the picture above) and browse to a folder on your drive to store the project. Generally, you can save it in My Documents.

Browse For Folder	? ×
Please select the folder to create your project in	
	_
Desktop My Documents	-
E Q My Computer	
My computer	
Share-to-Web Upload Folder Solution	
·····································	
graduation 2005	-
Folder: My Documents	
OK Cancel <u>N</u> ew Fo	older

Press Ok after selecting My Documents

Console application		×				
🐻 Console	Please select the folder where you want the new project to be created as well as its title.					
	Project title: First Program					
	Folder to create project in: E:\Documents and Settings\goetz\My Documents					
	Project filename: First Program					
	, Resulting filename: \goetz\My Documents\First Program\First Program.cbp					
	< <u>Back</u> <u>N</u> ext > <u>C</u> ancel					

When the directory for your project has been selected, the system will return to the **Console application**. As shown, Code Blocks will create a directory called First Program (Project Title) and returns your selected directory in **Folder to create project in**. Inside that directory will be the **Project filename** (First Program) and a resulting filename, which contains a Code Block Project file (.cbp) named First Program.cbp. The project title and project filename in this case are the same. However, they need not be the same and these names can be altered. Click on the Next Button when done.

Console application	×
🐻 Console	Please select the compiler to use and which configurations you want enabled in your project.
	GNU GCC Compiler
	Create "Debug" configuration: Debug
	"Debug" options
	Output dir.: bin\Debug
	Objects output dir.: obj\Debug
	Create "Release" configuration: Release
	"Release" options
	Output dir.: bin\Release
	Objects output dir.: obj\Release
	< <u>B</u> ack <u>N</u> ext > <u>C</u> ancel

The Next window to pop up will be the Compiler screen. This specifies where the **Debug** and **Release** compiled versions of your program will be placed. Leave this setting alone and press Next.

Console application	×
Console	Please select the language you want to use.
	< <u>B</u> ack <u>F</u> inish <u>C</u> ancel

The next window allows you to choose the language that you will use. Select the language as C++, then press Finish.



The system will then return to the [First Program] window and you are ready to write your program. It should be noted that the Build target is **Debug**, which will allow you to use the debugger to find errors. In the **Management** area of the screen (Shift-F2 toggles the Management display), you will see the files that are part of the project in the **Projects** tab. To see the source files, click on the plus [+]'s to expand the **Workspace** and its subdirectories.



Under Sources, there is a file called main.cpp, which is automatically created for you when you build a console application.

Adding Files To Your Project

If you have a project with additional existing files, go to the Project menu and select "Add files." This will bring in the files associated with your program. You also have the option to **Remove files**, performing **Build options** and to **Set programs' arguments...**.

<mark></mark> [First Program] - Code:	:Blocks v1	.0						_ 🗆	×
<u>File E</u> dit <u>V</u> iew Search P	roject <u>B</u> uil	l <u>D</u> ebug	wxSmith	<u>T</u> ools	Plugins	<u>S</u> ettings	Help		
	Add files Add files ro Remove fil	ecursively. es		-	1	<u>~</u>			
፤ 💵 🖷 😚 🚠 🤭 🦿	Set progra		ents						
Management Projects Symbols	Build optio Properties	าร							
Overkspace → → → → → → → → → → → → →									
Open files list	×	lessages							×
Dpened Files	C E T	oading: A: IlassWizari opystrings invVars So ioDoList w:	e::Blocks StylePlugin d CodeCom s Debugger ource Expor xSmith wxS artup script	Autosav pletion (FilesExt ter Help mithMim	ve BYOGa CodeSnipj ensionHa Plugin cbł	mes CBProl pets Codes Indler DevP KeyBinder S	Statistics Co PakUpdater ScriptedWiz	ompiler DragScroll ard	
Add files to t									11.

Add files to proje	ct				<u>? ×</u>
Look jn:	🚖 First Program		•	+ 🗈 💣 🎟•	
History History Desktop My Documents My Computer	First Program.cb	Ρ			
	File <u>n</u> ame:			•	<u>O</u> pen
My Network P	Files of <u>type</u> :	All files (*.*)		•	Cancel

Clicking on **Add files to project**, will bring up a window so you can browse to where your files that you wish to add are. Select any additional file you want to add and press Open. The file will then be added to your project.

If you are creating a new file, you can use the pull-down File menu and open an empty file.

First Program] - Code::Blocks v1.0	
File Edit View Search Project Build Debu	g wxSmith <u>T</u> ools Plugins <u>S</u> ettings <u>H</u> elp
New 🕨	Empty file Ctrl-Shift-N
📬 Open Ctrl-O	Project
Open default workspace	Build target
Recent projects	File
Recent files	Custom
- Terrent and the	From user template
Import project	
🔚 Save Ctrl-S	
🔛 Save <u>a</u> s	
Save all files Ctrl-Shift-S	
Save project	
Save project as	
Save project as user-template	
Save all projects	×
Cause weeks and	le::Blocks 🚺 Code::Blocks Debug 🛛 🔹 🕨
Save <u>w</u> orkspace Save wor <u>k</u> space as	StylePlugin Autosave BYOGames CBProfiler CB_Koders
Close workspace	rd CodeCompletion CodeSnippets CodeStatistics Compiler
Close workspace	s Debugger FilesExtensionHandler DevPakUpdater DragScroll
S ⊆lose file Ctrl-W	ource Exporter HelpPlugin cbKeyBinder ScriptedWizard vxSmith wxSmithMime wxSmithWizard WindowsXPLookNFeel
Close all files Ctrl-Shift-W	cartup script
Close project	
Close all projects	

File, New, Empty file.

You will be asked if you want to add this file to the project.



Choose Yes.

Code Blocks will ask for a file name to save the file as:

Save file					? ×
Save jn:	🔁 First Program		•	+ 🗈 💣 🎫	
History	C main.cpp				
Desktop My Documents					
My Computer					
My Network P	File <u>n</u> ame:	sample.cpp		•	<u>S</u> ave
My Network P	Save as <u>t</u> ype:	C/C++ files		•	Cancel

Give a name to the file. Pick a name that is related to the content of the file. Here it is called sample.cpp. C++ files need to be of the type cpp. Press Save to save the file.

Multiple selection Select the targets this file should belong to:	_ _ ×
♥ Debug ♥ Release	Wildcard select Toggle selection Select All Deselect All Selected: 2
OK Cancel	

Press Select All to have this file saved as both Debug & Release targets. Press OK when done.

A target is a type of compiled version. You can work with a debug target, which will allow you to test the program using a debugger. A debug target will be large in size, because it has extra information in it to allow you to test for errors. A release target is smaller in size, because it does not have the debugging information. When you are ready to give other people (such as your Instructor) your finished program, you should give them the release target.

sample.cpp [First Program]	- Cod	e::Blocks v1.0							미꼬
<u>File E</u> dit <u>V</u> iew Sea <u>r</u> ch <u>P</u> roject	<u>B</u> uild	d <u>D</u> ebug wxSmith	<u>T</u> ools	Plugins	Setti	ings	Help		
i 🗈 🛤 🖨 🔦 🔖 🚽	ŕÞ	🛍 🔍 🕓							
<u>.</u>					7				
🕴 😒 🍃 🧊 🐼 🛛 Build targ	et: De	bug	-]					
i 📲 😤 🕃 🤔 🚳 🛛		i.							
Management	×	sample.cpp						4	⊳ × ∣
Projects Symbols 4	▶∣Г	1							
- Workspace		2							
🚊 💾 First Program	- 11								
🖻 🔁 Sources	- 11								
main.cpp	- 11								
sample.cpp	- 11								
	- 10	•							
Open files list	×	Aessages							×
Dened Files		🚺 Code::Blocks	0	ode::Bloc	ks Deb	oug		<	1 🕨
		.oading: AStylePlugin	Autosav	ve BYOGa	mes C	BProfi	ler CB_Kod	lers	
		lassWizard CodeCon							. –
		opystrings Debugger NyVars Source Expo							
		oDoList wxSmith wxS							
		Running startup script							
E:\Document default	Line 1	I, Column 1		Insert			Rea	d/Write	1

The Sources now has sample.cpp as a source file in addition to the main.cpp file.

Since the sample.cpp is not needed for your project, please remove it.

sample.cpp [First Progra	am] - Code::Bloc	ks v1.0						×
File Edit View Search Pr	oject Build Deb	ug wxSmith	Tools	Plugins	Settings	Help		
i 🗋 🖆 🛗 📥 👘	Add files							
-	Add files recursive Remove files	ly			~			
			-	1				
↓ ● ₩ ۞ ₨ ? /	Project tree Set programs' arg	· manta						
		unencs						
Management	Build options							×
Projects Symbols	Properties	2						
E First Program		-						
E Sources								
main.cpp								
sample.cpp								
Open files list	× Message							×
🔁 Opened Files			12/2	uda u Dia d	a Dahua	1	4	, î
Untitled2		ode::Blocks				Files CD. Kada	N	-
	ClassWi	AStylePlugin ard CodeCom	pletion (IodeSnipp	pets Code:	Statistics Con	npiler	
		ngs Debugger Source Expor						
	ToDoList	wxSmith wxS						
	Kunning	startup script						•
Remove files default	Line 1, Colur	nn 1		Insert		Read	l/Write	//.

From the Project menu select, Remove files.

Multiple selection	<u>_ </u>
Select files to remove from First Program:	
main.cpp	Wildcard select
sample.cpp	Toggle selection
	Select All
	Deselect All
	Selected: 1
OK Cancel	

Place a check mark next to any file(s) that you wish to remove. Press OK when you are done.



You will need to confirm that you wish to remove the file(s). Press Yes, if you are sure you want to remove them. Otherwise press No.



You will now see an updated listing of the Sources in your file. You should now see only Main.cpp. In the Open Files list, there may be a file called **!Untitled. Please ignore this.**

To edit a file from your project, double click on it's name from **Sources** and it will appear in the window with line numbers. You can now edit the file and prepare your program.





In order to check that Debug is running, you can use the **Project** pull-down menu and click on **Build Options.**

Project build options		
Project build options	Selected compiler GNU GCC Compiler Compiler Linker Directories Commands Custom variables "Make" commands Policy: Append target options to project options Compiler Flags Other options Compiler Flags Other options #defines Categories: Categories: Categories: Categories:	
	OK Cancel	

When this is done, the **Project Build options** window will come up. Make sure that the **Produce debugging symbols [-g]** is checked.

Press OK when done.



After clicking on done, the system will return to Main.cpp. When testing your code, make sure that **Debug is selected as the target to use.** This way when you **Compile** your program, you will have a Debug version available. To compile a file means to take the instructions that you have written and translate it into machine code for the computer to understand.


Compile your file from the Build pull-down menu by clicking on Compile current file (Ctrl-Shift-F9).



Test the project from the Build Pull-down menu, by clicking on Build and Run. This step will **build** an executable file for you. A project build will take the compiled versions of your source files and combine them into one program.

You are able to press F9, which is a keyboard shortcut that will build your project and run it at the same time. As you gain more experience with the system, it will be easier to just press F9 to Build & Run your program. The Message window will indicate if there are any errors during a compile or build phase.



This is the output from my first program. Notice that besides displaying "Hello world!" it also says to "Press any key to continue" with the program paused. Pressing any key will exit the program.

If you execute the program by going to a console window you will not see the "Press any key to continue" message:



Notice that there are double quotes around the file name. This is because there is a space in the name. If you execute this program by double clicking on it's icon, the program would close right away. That is because the pause statement is only done when you run your program in Code Blocks.

main.cpp [First Program] - Code::Blocks v1.0 - 🗆 🗵 File Edit View Search Project Build Debug wxSmith Tools Plugins Settings Help R New. ۲ main() : int 🚞 Open Ctrl-O Open default workspace • Recent projects ۶ Recent files Þ $\triangleleft \rightarrow \mathbf{X}$ Import project include ≺iostream≻ 🔚 Save Ctrl-S 👩 Save <u>a</u>s... nt main() Ctrl-Shift-S Save all files std::cout << "Hello world!" << std::endl;</pre> return 0; Save project Save project as... Save project as user-template ۲ Save all projects × Save workspace 🚫 Debugger 4 \triangleright 🔶 Build messages Save workspace as... . . ٠ Close workspace kpoints e and version: GNU gdb 6.3 🖸 <u>⊂</u>lose file Ctrl-W ~1/goetz/MYDOCU~1/FIRSTP~1/main.cpp:5 ished with status O Close all files Ctrl-Shift-W ▼ Close project 11 Insert Read/Write Close all projects

When you are done, save all your files by pulling down the File menu and clicking on Save all files.

	v1.0
<u>File</u> Edit <u>V</u> iew Search <u>P</u> roject <u>B</u> uild <u>D</u> eb New	ug wxSmith <u>T</u> ools Plugins <u>S</u> ettings <u>H</u> elp • <mark>K R</mark>
Open Ctrl-O Open default workspace <u>R</u> ecent projects <u>R</u> ecent files	main() : int
Import project Save Ctrl-5 Save as Save all files Ctrl-Shift-S	<pre>include <iostream> int main() std::cout << "Hello world!" << std::endl;</iostream></pre>
Save project Save project as Save project as user-template Save all projects	return 0;
Save <u>w</u> orkspace Save wor <u>k</u> space as Close workspace	Points e and version: GNU gdb 6.3
♀ ♀ ♀ ♀ <	<pre>~1/goetz/MYDOCU~1/FIRSTP~1/main.cpp:5 ished with status 0 11 Insert Read/Write</pre>

Now you can select to save the project:

When you exit the program, you may be asked to save the **Workspace** and the **Layout**. The Layout refers to the placement of various windows that you may have positioned. Generally you would select to Save the Layout (unless you know you really do not want it saved). The Workspace refers to the projects you are working on. It is possible for you to be working on multiple projects within your workspace. Saving your workspace will allow you to return to the same set of projects when you next open Code Blocks.

To open a project

Code::Blocks v1.0		
<u>File E</u> dit <u>V</u> iew Sea <u>r</u> ch <u>P</u> roje	ct <u>B</u> uild <u>D</u> ebug	g wxSmith <u>T</u> ools Plugins <u>S</u> ettings <u>H</u> elp
New	•	s B
🔁 Open	Ctrl-O	T
Open default workspace		
Recent projects	•	
<u>R</u> ecent files	•	
Import project	•	4 b x
🗐 Save	Ctrl-S	-
🛃 Save <u>a</u> s		
Save all files	Ctrl-Shift-S	Code::Blocks 🥤
Save project		
Saye project as		The open source, cross-platform IDE
Save project as user-template		http://www.codeblacks.opg
Save all projects		
Save <u>w</u> orkspace		n IXA Cadau Blacks Dahun I O. Saawah yaawiha I A. N
Save workspace as		ks 🛛 🔌 Code::Blocks Debug 🛛 🖓 Search results 🛛 4 🕨
Close workspace		
😧 <u>C</u> lose file	Ctrl-W	
Close all files	Ctrl-Shift-W	_
Close project		
Close all projects		

From the File menu select Open.

Open file		<u>? ×</u>
Look <u>i</u> n:	: 🔄 First Program 🔽 🖛 🗈 📸 🎫	
History Desktop My Documents	🚘 obj 🗲 First Program.cbp	
My Computer	File name: Image: Code::Blocks project files	<u>O</u> pen Cancel

From the **Files of type:** in the window, select "Code::Blocks project files" and then select the .cbp file pertaining to your program.

Press Open when done.



The project has reopened. You can get more space to see your program, if you close the **Messages** window. Pressing **F2** toggles the display of the messages. The **Messages** window has been turned off for the remainder of this tutorial, to allow more space to be visible on the screen.

Debugging a Program

As your programs become more complicated, there will be a need to trace the program execution step by step or place break points where you wish the program to pause. This is where a debugger is utilized. A debugger can pause your program and you can watch the values of the variables that you have defined.

The following is a sample program that can be traced "line by line" while watching what happens as each line of code is executed.



First, it is necessary to set a place in the code to have the program pause. This is done by using the **Debug** pull-down menu and clicking on **Run to Cursor. The cursor should be over the first line of code** where you wish to start the tracing process. This starts the debugging process.

📙 main.cpp [debug] - Code::B	locks v1.	.0		
	_	Debug wxSmith Tools	Pļugins <u>S</u> etting	as <u>H</u> elp
i 🖻 🖀 🖨 🗞 🖓 i	<u>* 6 </u>	_	F8	
		😡 Stop debugger		nain() : int
i 😒 👂 🧊 🐼 Build tar	rget: Debu	Continue	Ctrl-F7	ት. ኛን 😗 🙆 🗔 🖡
Management X	main.ct	😚 Next line	F7	4 Þ ×
Projects 4 🕨	1	Next instruction	Alt-F7	
⊡ 😡 Workspace	2 3	矜 Step into	Shift-F7	
e debug	4	😚 Step out	Ctrl-Shift-F7	
main.cpp	5 6	Toggle breakpoint	F5	
	7	Run to cursor	F4	
	8 9	Add symbol file		
	10	Debugging windows	+	
	11	Information	+	
Open files list 🛛 🗙	12 13	Edit <u>w</u> atches		
🔁 Opened Files	14	Attach to process		
i 🧾 main.cpp	15 16	Detach		
	17 18	Send user command to	o debugger	
	19	cout << z	<< endl;	-
	20 21	return 0;		
	22	L)		
	23			
	Line 1/	Column 10	Terrent	Daadittuita
Run program WINDOWS-1252	Line 1	3, Column 18	Insert	Read/Write

The next step in debugging a program is to tell the program when to stop running so you can inspect the results. To do this, place the cursor over the line where you want your program to stop. For example, the cursor was placed at line 18 (which is hidden behind the menu). This is called a **Breakpoint.** Now you can instruct the debugger to run the program up to the cursor's position (line number).



The program will generate a blank window. It is blank, since that program has yet to execute any line that displays something.

- Code::Blo	ocks v1.0		
File Edit View Search Project	Build Debug wxSmith Tools	Plugins Settin	gs Help
i 🗋 🖆 🚔 📥 🔦 🔖 🦂	🚹 🚺 🚛 Start	F8	
	🔀 Stop debugger		nain() : int
🕴 🐤 💊 🤝 🐼 🛛 Build targe	et: Debi 🚛 Continue	Ctrl-F7	🚠 ኛን 🚱 🗔 🚛
Management ×	main.cr 🕜 Next line	F7	4 b ×
Projects 4 🕨	1 💦 Next instruction	Alt-F7	
🖃 😡 Workspace	2 🔥 Step into	Shift-F7	
e ebug	3 4 Step out	Ctrl-Shift-F7	
i main.cpp	5 6 Toggle breakpoint	F5	
	7 🧏 Run to cursor	F4	
	9 Add symbol file		
	10 Debugging windows	•	Breakpoints
	11 Information	•	• Call stack
Open files list ×	13 Edit <u>w</u> atches		CPU Registers
Dened Files	14 Attach to process 15 Detach		Disassembly Examine memory Running threads
	17 18 Send user command to (debugger	Watches
	19 cout << z <	< endl;	
	20 21 22 23 return 0; 23		
Watch variab WINDOWS-1252	Line 18, Column 1	Insert	Read/Write

To watch certain variables during the execution of the program, you should open the Watches window. This will show you the variables in your code. This is accomplished by going to the **Debug** pull-down menu and clicking on **Debugging Windows** and then **Watches**.



These are the watches that the debugger is displaying. Notice that $\times \&_{y}$ have the correct values. Variable z has not been assigned it's value on line 18 yet. The current value is a random value.



Line 18 has a yellow marker on the left side. This indicates that the program has paused on that line, which is the breakpoint.

📕 main.cpp [debug] - Code::B	locks v1.0	D		
File Edit View Search Project		Debug wxSmith Tools ↓ Start	Plugins <u>S</u> ettin F8	ngs <u>H</u> elp
:		🐼 Stop debugger		nain() : int
i 😒 👂 🦚 🐼 🛛 Build tar	get: Debi	Continue	Ctrl-F7	🚠 ኛን 😚 🖬 🛄 ∔
Management X		😚 Next line	F7	4 Þ ×
Projects 4 >		Next instruction	Alt-F7	
e debug	3 -	Step into Step out	Shift-F7 Ctrl-Shift-F7	
i 🗋 main.cpp		Toggle breakpoint	F5 F4	
Open files list X	8 9 10 11 12 13	Add symbol file Debugging windows Information Edit <u>w</u> atches		> >
Dened Files	14 15 16	Attach to process Detach		
	17 18 _	Send user command to	debugger	
	19 20 21 22 23	cout << z return 0; }	<< endl;	Þ
Execute the r WINDOWS-1252	Line 18	, Column 1	Insert	Read/Write

To determine how your program will function when calling functions such as:

z=addem(x,y);

Step info (Shift-F7) can be selected from the Debug pull-down menu.



The next step is line 9.



The arguments a and b are shown in the Watches window. Notice that the local variable c, which has not been set yet, has a random value.

📕 main.cpp [debug] - Code::Blocks v1.0				
File Edit View Search Proje	_	Debug wxSmith Tools	P <u>l</u> ugins <u>S</u> etting F8	
		Continue	Ctrl-F7	∴ ↔
Management X Projects 4 Projects 4 Projects 4 Open files list X Open files list X Open files list X	1 2 3 4 5 6 7 8 9 10 11 12 13 14	Next line Next instruction Step into Step out Toggle breakpoint Run to cursor Add symbol file Debugging windows Information Edit watches Attach to process	F7 Alt-F7 Shift-F7 Ctrl-Shift-F7 F5 F4	
Execute the r WINDOW5-1252	15 16 17 18 19 20 21 22 23 Line 9	Detach Send user command to cout << z return 0; } Column 1	-	Read/Write

To proceed to the next line of code, select Next line from the Debug menu.

Pressing **F7** is a useful keyboard shortcut and will become second nature as you become familiar with the system.

Watches	×
E-Local variables	
c = 7	- 1
E Function Arguments	- 1
- a = 5	- 1
b = 2	- 1
	- 1
	- 1
	- 1
	- 1
	- 1
	- 1
	- 1
L	

The debug window reflects the change of c.

main.cpp [debug] - Code::Blocks v1.0				
File Edit View Search Project	_	Debug wxSmith Tools ↓ Start Stop debugger	Plugins <u>S</u>ettin g F8	gs Help addem(int a, int b) : int
i 💊 🖒 🦚 🐼 🗵 Build tar Management 🗙			Ctrl-F7	ት 🖓 🖑 😡 🗔 🛵
Projects 4 	main.cr 1 2 3 4	 Next line Next instruction Step into Step out 	F7 Alt-F7 Shift-F7 Ctrl-Shift-F7	
main.cpp	5 6 7 8 9	Toggle breakpoint Run to cursor Add symbol file	F5 F4	-
Open files list X	10 11 12 13	Debugging windows Information Edit <u>w</u> atches) }	
main.cpp	14 15 16 17	Attach to process Detach Send user command to o	debuager	-
	18 . 19 20 21 22 23	<pre>cout << z < return 0; }</pre>		
Continue exe WINDOWS-1252	Line 11	., Column 1	Insert	Read/Write

When you are done debugging, you can click on **Continue and your program will run to completion.** This is better than selecting to **Stop debugger**. The reason it is better to **Continue**, is because the program comes to a natural end, rather than aborting. However if your program is stuck in a loop, or you are sure you can exit safely, you can select from the Debug menu **Stop Debugger**.

<mark>–</mark> main.cpp [debug] - Code::Bloo	cks v1.0		
	Build Debug wxSmith Tools	Plugins <u>S</u> ettings F8	Help
🔢 🕞 💊 🐼 🐼 Build targe	t: Debi JF Continue		ldem(int a, int b) : int 扰 🏞 (♂ 🗵 🛄 🚛
Management X Projects 4 Workspace	Tain.cr The section 1 The section 2 The section	F7 Alt-F7 Shift-F7	4 ▷ ×
debug	3 4 Step out 5 Toggle breakpoint	Ctrl-Shift-F7	
Open files list ×	7 Part Run to cursor 8 Add symbol file 10 Debugging windows 11 Information 12 Edit watches	F4	
Dened Files	14 Attach to process 15 Detach 16 17 18 Send user command to		
	19 cout << z 20 21 return 0; 23	<< endl;	
Toggle break; WINDOWS-1252	Line 9, Column 9	Insert	Read/Write

You can further define places in your program to pause and allow you to inspect the code. This is done by setting breakpoints in your code. You can have zero or more breakpoints in your code. When the debugger encounters a breakpoint, the program pauses and the debugger will allow you to inspect your code. The breakpoint remains until you remove it. It can be Toggled with F5.

		1
main.cpp [debug] - Code::		×
<u>File Edit View Search Proje</u>	ct <u>B</u> uild <u>D</u> ebug wxSmith <u>T</u> ools Plugins <u>S</u> ettings <u>H</u> elp	
E 🖻 🖬 🖨 🔦 🔖	💰 🗗 💼 🔍 🕓	
:	addem(int a, int b) : int	
i 🤝 👂 🧊 🐼 Build ta	rget: Debug 💽 🕴 📲 😵 🔝 🥐 😂 🗔	i.
Management ×	main.cpp d b >	¢
Projects 4 🕨	1 #include <iostream></iostream>	
🖃 😡 Workspace	2 using namespace std;	
🗄 💾 debug	3	
🖻 🗁 Sources	<pre>4 int addem(int, int);</pre>	
main.cpp	5 6 ⊡int addem(int a, int b) (
	7 int c;	
	8	
	9 🔶 c=a+b;	
	10	
	11 return c;	
Open files list 🛛 🗙	13	
🔁 Opened Files	14 int main()	
i 📄 main.cpp	15 🕀 (
	16 int x=5, y=2, z;	
	17	
	$\begin{array}{c c} 18 & z = addem(x, y); \\ 19 & cout << z << endl; \\ \end{array}$	
	20	
	21 return 0;	
	22 L)	
	23	
		۲
E:\Document: WINDOWS-1252	Line 9, Column 11 Insert Read/Write	//

A breakpoint has been set at line 9. The red circle indicates that there is a breakpoint in the code.

🕂 main.cpp [debug] - Code::Blocks v	1.0		
<u>File E</u> dit <u>V</u> iew Sea <u>r</u> ch <u>P</u> roject <u>B</u> uik	Debug wxSmith Tools	Plugins <u>S</u> etting	is <u>H</u> elp
i 🗋 🖬 🖨 🔦 🔖 🚜 🗗	 ↓] <u>S</u> tart	F8	
	🔁 🔀 Stop debugger		addem(int a, int b) : int
🕴 💊 🜔 🧊 🐼 🖉 Build target: De	Continue	Ctrl-F7	🚠 🤭 🥙 🗔 💰
Management × main.	🛊 🏠 Next line	F7	4 b x
Projects 4 >	🚯 Next instruction	Alt-F7	
Workspace		Shift-F7	
eres		Ctrl-Shift-F7	
main.cpp	Togglo brookpoint	F5	
	📲 Run to cursor	F4	
1	Debugging windows	•	
1.	Information	•	
Open files list X 1: Opened Files 1:			
main.cpp 1.	Attach to process		
10			
1'	Send user command to	debugger	
1:		<< endl;	
2.			
2:			
Run current p WINDOWS-1252 Line	9, Column 11	Insert	Read/Write

The program is started by selecting from the **Debug** pull-down menu, **Start**. This will run the program in the debugger until a breakpoint is encountered, at which point the program will pause.

- Code::l	Blocks v1.0							
Eile Edit View Search Project Build Debug wxSmith Tools Plugins Settings Help								
i 🗋 🖼 🚔 🔦 🔖 🦓 🖆 🏦 💁 🕵								
addem(int a, int b) : int								
📔 🕒 💊 🐼 🐼 Build target: Debug 🔄 🔄 🖬 🐨 🙃 🕐 (* 😣 🗔 🚛								
Management X	main.cpp	 4						
Projects 4 🕨	1 #in	clude <iostream></iostream>						
🖃 🞧 Workspace	2 usi	ng namespace std;						
📄 💾 debug	з							
🖻 🦻 Sources	4 int 5	addem(int, int);						
main.cpp	-	addem(int a, int b) (
	7	int c;						
	8							
	9 🗘	c=a+b;						
	10	return c;						
	12	recur c,						
Open files list X	13							
Opened Files		main()						
····· 📄 main.cpp	15 🖂 (
	16 17	<pre>int x=5, y=2, z;</pre>						
	18 z=addem(x,y);							
	19 cout << z << endl;							
	20							
		21 return 0; 22						
	23							
	4							
E:\Document: WINDOWS-1252	Line 9, Column 1	Insert Read/Write						
Le de comence l'astrano da parte 252	June 9, Coldmin 1	priserc j pread/write //						

When the program pauses at the break point, a red circle with a yellow triangle mark will appear at the breakpoint.

hain.cpp [debug] - Code::Blocks v1	.0		
<u>File E</u> dit <u>V</u> iew Sea <u>r</u> ch <u>P</u> roject <u>B</u> uild	Debug wxSmith Tools	Plugins <u>S</u> etting:	s <u>H</u> elp
🗋 🖆 🚔 📥 🔦 🍡 🖓 🗗	↓ <u>S</u> tart	F8	
1	😣 Stop debugger		nain() : int
🕴 😒 🜔 🦚 🐼 🖉 Build target: Debu	↓ 📔 Continue	Ctrl-F7	ት. 🏷 🖑 😆 🗔 🚛
Management × main.cp	😚 Next line	F7	4 Þ ×
Projects 4 1	😚 Next instruction	Alt-F7	
Workspace	Step into	Shift-F7	
debug 3	🔥 Step out	Ctrl-Shift-F7	
main.cpp 6	Toggle breakpoint	F5	
7	🕌 Run to cursor	F4	
8	Add symbol file		
10	Debugging windows	+	
11	Information	•	
Open files list × 13	Edit <u>w</u> atches		
main.cpp 14	Attach to process		
16	Detach		
17	Send user command to	debugger	
19 20	cout << z	<< endl;	
20 21	return 0;		
22	L)		
			Þ
Toggle break WINDOWS-1252 Line 1	9, Column 1	Insert	Read/Write

You can set multiple breakpoints. The keyboard shortcut **F5** allows you to toggle the breakpoint at any line.



This screen shows breakpoints on lines 9 and 19, but line 9 indicates that the code has executed to that point.

main.cpp [debug] - Code::Blocks v1.0								
File Edit View Search Project		Debug wxSmith Tools	Plugins <u>S</u> etting F8					
i 😒 🕨 🧊 🐼 Build ta	rget: Debi	Continue	Ctrl-F7	nain() : int 🙃 🏷 🖑 🙆 🗔 🛵				
Management X Projects Workspace Workspace Sources main.cpp Open files list X	main.cr	 Next line Next instruction Step into Step out Toggle breakpoint Run to cursor Add symbol file Debugging windows Information Edit watches 	F7 Alt-F7 Shift-F7 Ctrl-Shift-F7 F5 F4					
Captions on UKUDOUUS 1953	14 15 16 17 18 20 21 22 23 ◀	Attach to process Detach Send user command to a coupt << z < return 0; }	< endl;					
Continue exe WINDOWS-1252	Line 19	9, Column 8	Insert	Read/Write				

Selecting Continue from the Debugger menu will run the program till the next breakpoint.



Now the program stops at line 19, because the program reached the second breakpoint. Press Ctrl-F7 to continue. Now the program runs till the end of the program, because there are no further breakpoints to encounter.

When you exit Code:Blocks you may be presented with the following window.



Say "Yes" to save the Workspace. This saves settings of the workspace you are working on.

You now know the basics of how to use the compiler, work with a project, and use the debugger. Good luck with your programming!