

Code::Blocks for C/C++ Novice

Chipset

Copyright (C) 2008, 2009, 2010 Chipset

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

目 录

封皮.....	1
目录.....	2
前言.....	3
1. 安装Code::Blocks.....	4
1.1 下载.....	4
1.2 安装.....	4
2. Code::Blocks的编程环境配置.....	8
2.1 环境.....	8
2.2 编辑器.....	13
2.3 编译器和调试器.....	14
3. 编写程序.....	18
3.1 创建一个工程.....	18
3.2 添加和删除文件.....	21
3.3 编辑文件.....	25
3.4 编译程序.....	29
3.5 调试程序.....	43
3.6 阅读别人编写的程序.....	70
3.7 代码性能测试.....	73
3.8 代码统计.....	76
4. 附录.....	77
4.1 Linux下安装Code::Blocks.....	77
4.2 Mac OS X下安装Code::Blocks.....	77
4.3 Code::Blocks搭配高版本gcc编译器.....	77
4.4 安装配置boost.....	79
GNU Free Documentation License.....	83
封底.....	89

前言

用高级计算机语言，例如C、C++，编写的程序，需要经过编译器编译，才能转化成机器能够执行的二进制代码。然而，把头脑中的思想转变成能够正常工作的计算机程序需要付出一定的努力和时间，因为为了让程序能够达到我们想要的结果，我们往往需要反复修改代码。本书的目的是帮助初学者学习组织程序编码逐步隔离并发现程序中的逻辑错误。通过本书，您可以学会怎么一步步的跟踪代码，找到问题出在什么地方，搞明白为何您的程序不能正常运行，这个过程称谓调试程序。手工跟踪能够有效的帮助初学者找到bug出在什么位置，消除bug，让程序正常运行。自动化的工具同样也能够帮助您跟踪程序，尤其当程序很复杂时效果更加明显，这种工具叫做调试器。调试器能够让运行中的程序根据您的需要暂停，查看程序怎么运作的。有些调试器是以命令行的形式工作的，较新的调试器有些具备好的图形界面，调试器能够方便的帮助您看到您定义的变量状态。基于图形界面的调试器是集成开发环境(IDE, 即Integrated Development Environment)的一部分。本书的作用就是帮助您学习使用这种环境以便更好的掌握编程技巧。

一个调试器并不能解决您程序中出现的的问题，它仅仅是一种帮助您编程的工具。您首先应该运用您手中的纸和笔分析程序，搞清到底怎么回事，一旦确定错误大致出在什么位置，便可以用调试器观察您的程序中特定变量的值。通过观察这些代码，可以了解到您的程序是怎么一步步执行的。

C/C++的IDE非常多，对于学习C/C++语言的朋友而言，用什么IDE可能并不重要，重要的是学习C/C++语言本身，不过，会用一款自己习惯的IDE进行程序的编写和调试确实很方便。

本书主要论述一款开源、免费、跨平台的集成开发环境Code::Blocks的安装、配置、以及程序的调试和编译等。Code::Blocks支持十几种常见的编译器，安装后占用较少的硬盘空间，个性化特性十分丰富，功能十分强大，而且易学易用。作者这里介绍的Code::Blocks集成了C/C++编辑器、编译器、和调试器于一体，使用它可以很方便的编辑、调试和编译C/C++应用程序。Code::Blocks具有很多实用的个性化特性，这里只会简单介绍少数几个常用的特性。

作者希望本书能够帮助您体验编程的乐趣的同时也能帮助您提高调试和编写程序的基本功。

如欲了解更多有关Code::Blocks的信息，请访问Code::Blocks的官方网站<http://www.codeblocks.org>。

1. 安装Code::Blocks

1.1 下载

为了安装Code::Blocks IDE，首先需要下载它们。如果您使用的是Windows 2000 或 Windows XP 或 Windows Vista操作系统，从下面地址下载Code::Blocks8.02(作者写本书时，8.02是最新的版本)这个IDE：

<http://downloads.sourceforge.net/codeblocks/codeblocks-8.02mingw-setup.exe>

以上地址下载的文件中包含了MinGW它(内嵌了GCC编译器和gdb调试器)

如果您仅仅希望把Code::Blocks当作编辑器使用，或者打算自己配置编译器和调试器的话，可以下载不带MinGW的版本，到下面的地址去下载。

<http://downloads.sourceforge.net/codeblocks/codeblocks-8.02-setup.exe>

本书的作者建议初学C/C++的朋友下载内置MinGW的版本，这样不致于花费太多时间配置编译器和调试器，从而把大部分时间用于学习调试和编写程序。待将来您熟悉了Code::Blocks，可以使用Nightly build的SVN版本，再搭配高版本的MinGW或者其它编译器一起使用。

如果您使用Mac OS X 或Linux操作系统，请参阅附录中的安装说明。

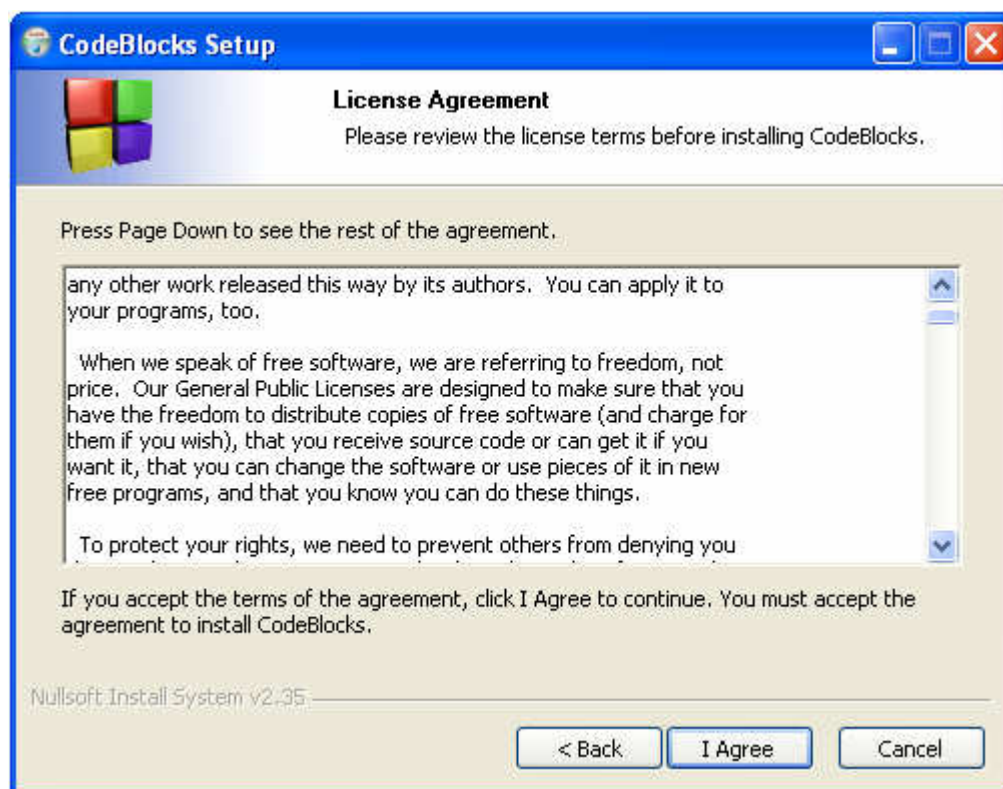
1.2 安装

安装过程可以参照如下步骤进行(以在笔者的英文版Windows XP Professional SP2操作系统上安装Code::Blocks8.02为例)。

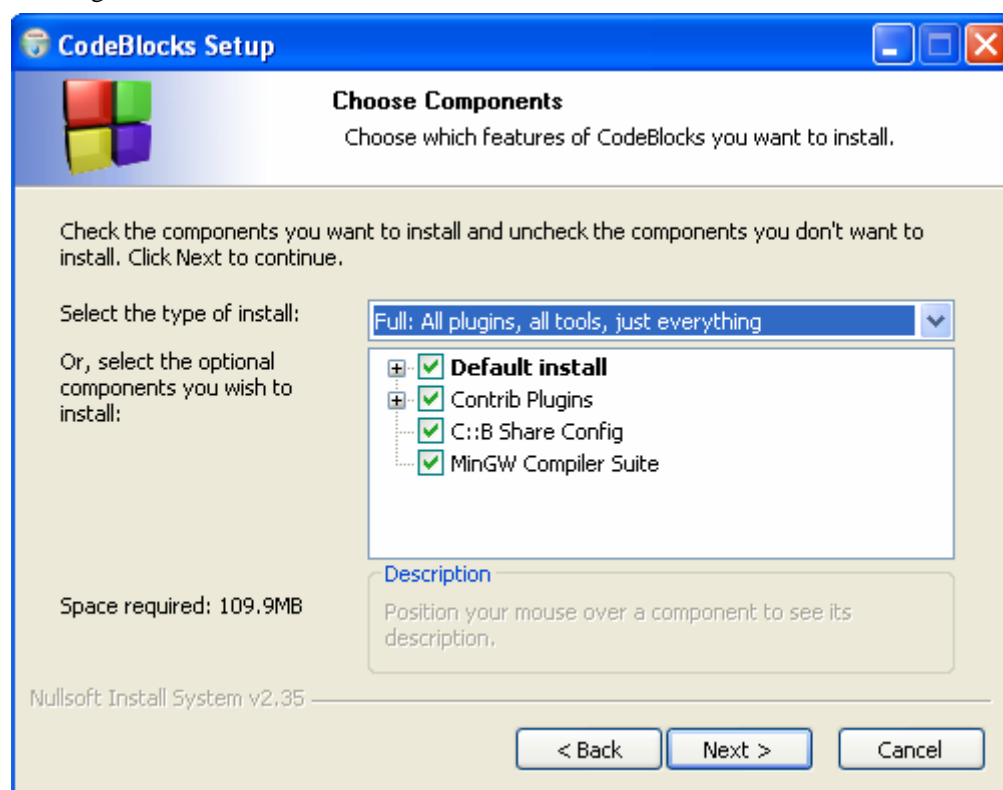
运行下载后的安装文件进入左下图界面。



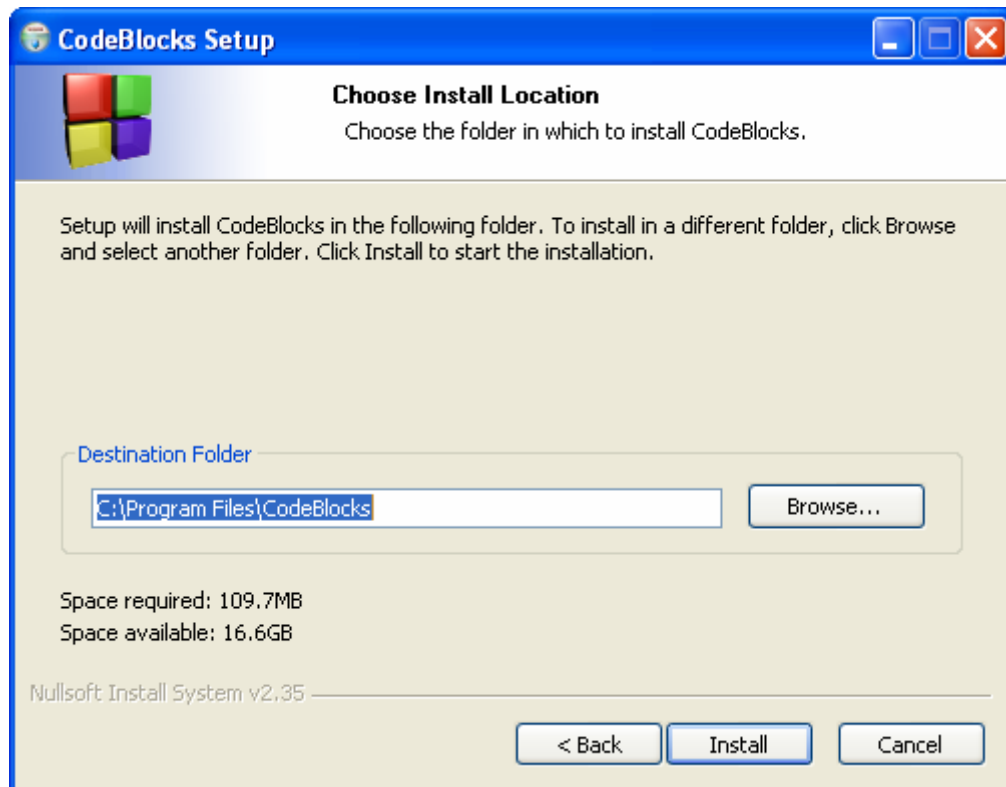
再用鼠标点击Next按钮，可以进入下图界面。



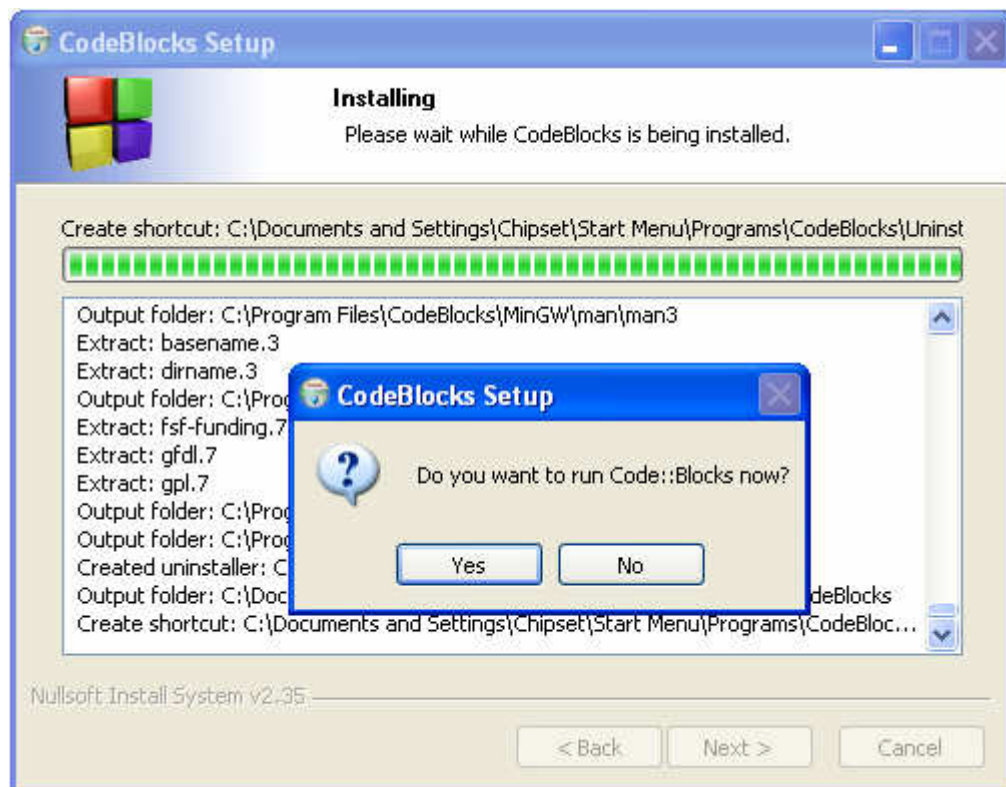
用鼠标选择 I Agree 按钮，进入如下图界面。



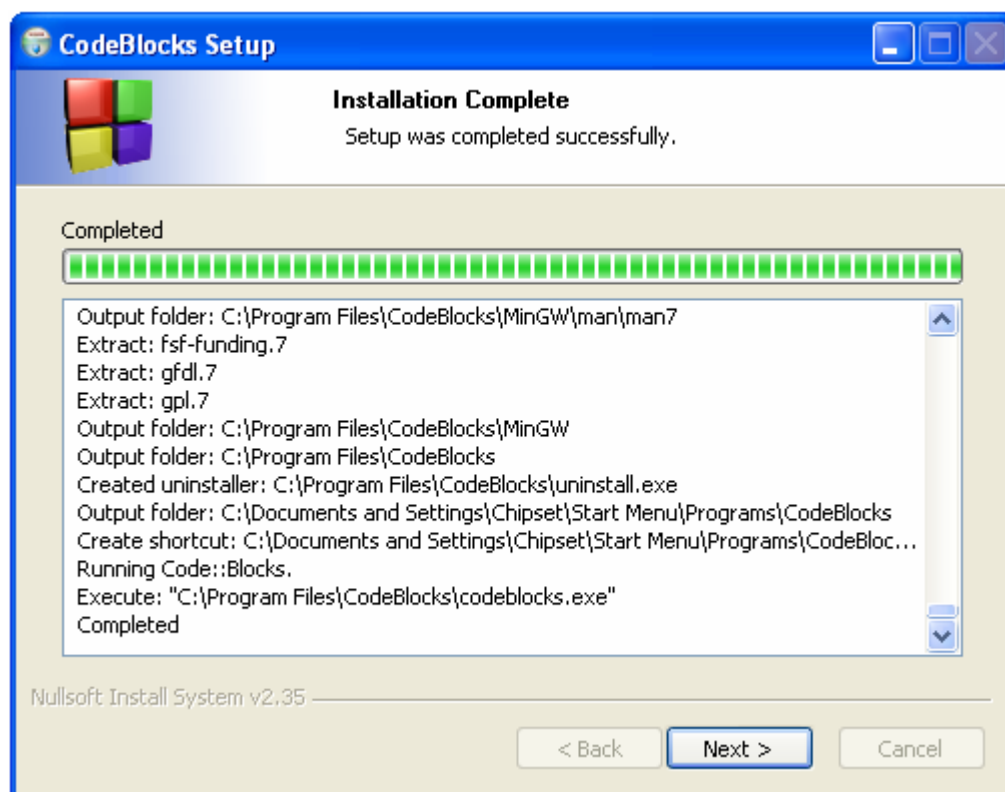
选择全部安装(Full: All plugins, all tools, just everything)，见上图，再点击 Next 按钮，进入一个新界面，如下图。



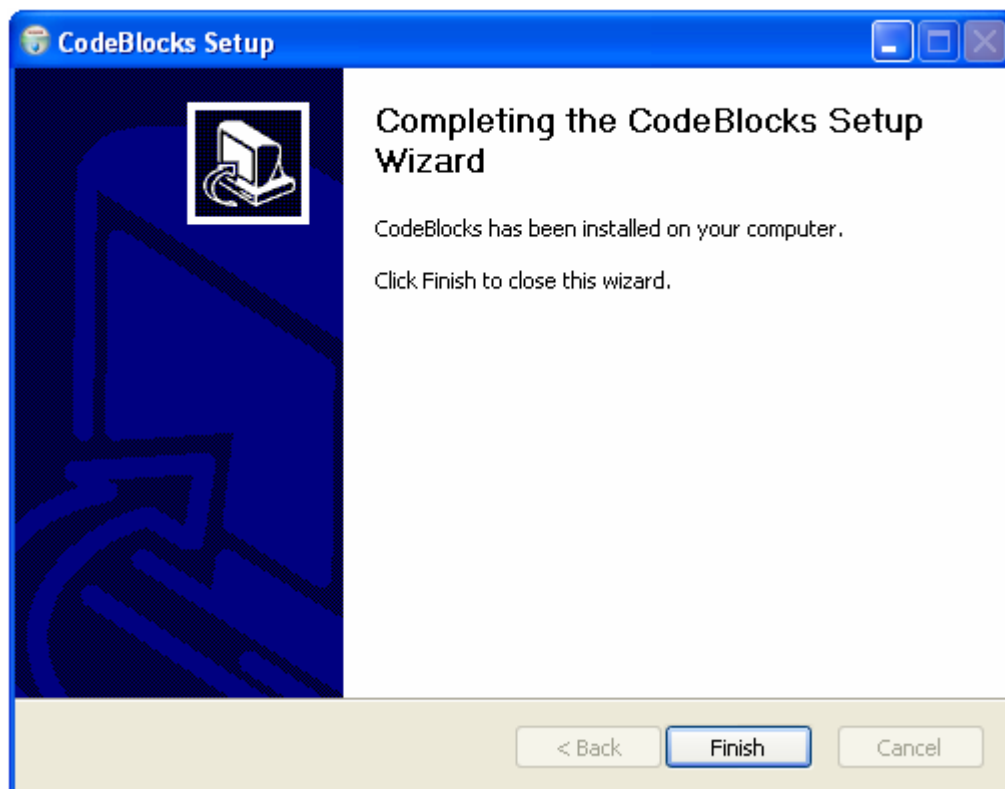
点击 Browse...按钮选好安装路径(默认安装路径为 C:\CodeBlocks), 用鼠标选择 Install 按钮, 可以看到安装过程正在进行, 并弹出一个对话框, 见下图。



用鼠标选择 No 按钮，则对话画框关闭，见下图。



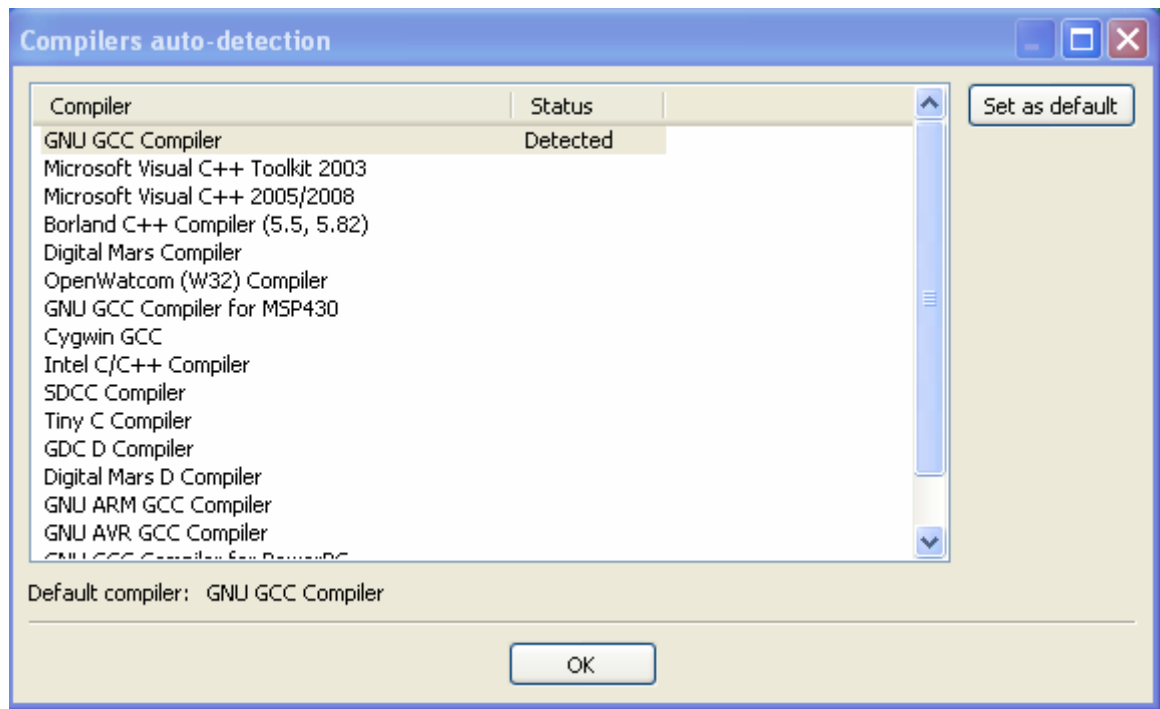
用鼠标选择 Next 按钮，进入界面见下图。



最后用鼠标选择 Finish，则安装过程就完成了。

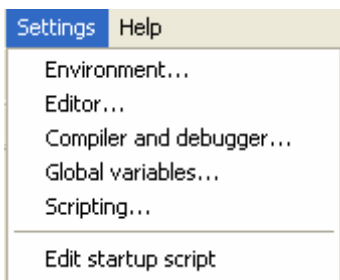
2. Code::Blocks 编程环境配置

第一次启动Code::Blocks，可能会出现如下对话框，告诉您自动检测到GNU GCC Compiler编译器，用鼠标选择对话框右侧的Set as default按钮，然后再选择OK按钮，见下图。

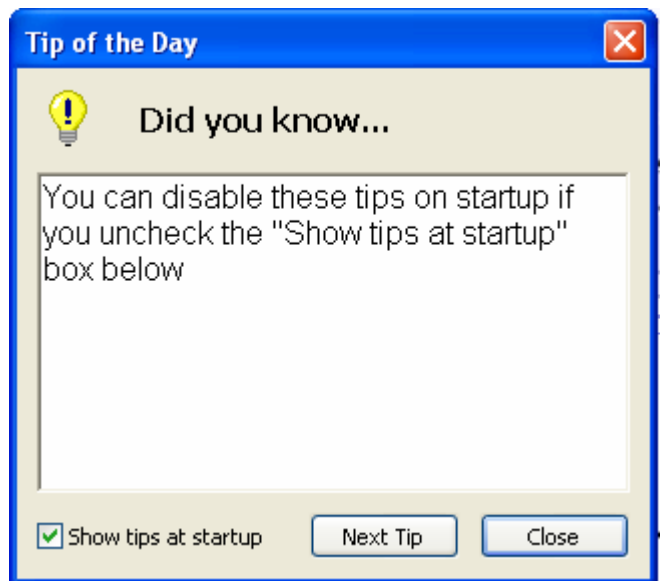


假如您的Code::Blocks安装正确的话，接下来就进入Code::Blocks的主界面，但是会弹出一个标签为Tips of the Day的小对话框，见右下图。

把Show tips at startup前面的勾去掉，然后选择Close，这样下次启动就不会再出现这个小对话框。



进入Code::Blocks主界面，选择主菜单Settings，弹出一个窗口，见左上图。然后我们就可以分别对环境(Environment...)，编辑器(Editor...)，编译器和调试器(Compiler and debugger...)三个子菜单进行配置了。



2.1 环境

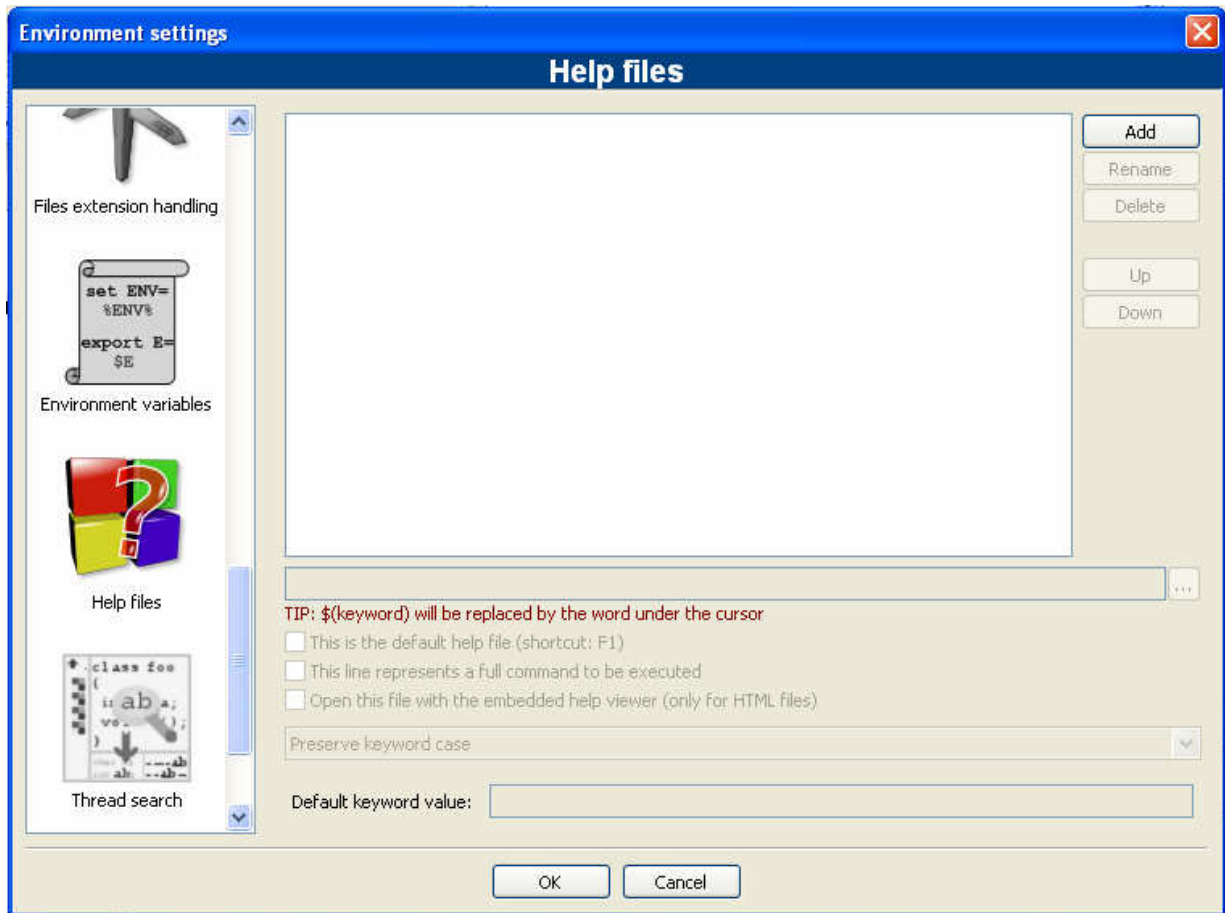
选择主菜单Settings下的第一个子菜单Environment...,会弹出一个窗口，用鼠标拖动左侧的滚动条，可以见到很多带有文字的图标。这些下面带有文字的图标代表了不同的功能按钮。

2.1.1 配置帮助文件

拖动滚动条，用鼠标选择这个图标，见右图。

此时会出现一个对话框界面，见下图。

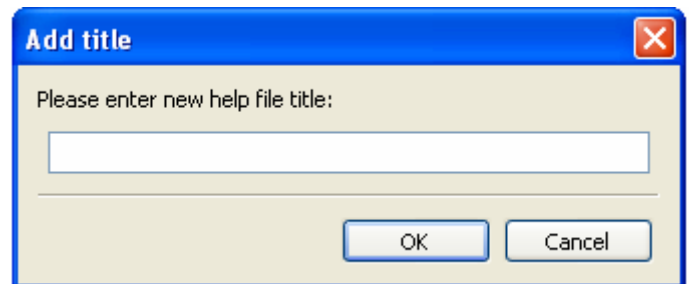




然后可以添加一些我们可能需要的帮助文件。我们编写基本的C/C++应用程序，仅需要知道C/C++的库函数用法就可以了。如果您没有C/C++语言库函数的文档，请到<http://www.cppblog.com/Files/Chipset/cppreference.zip>去下载C++ Reference，解压后放到Code::Blocks目录下(也可以放到别处)，以便添加进来编程时方便查阅。可以按照如下步骤进行添加：

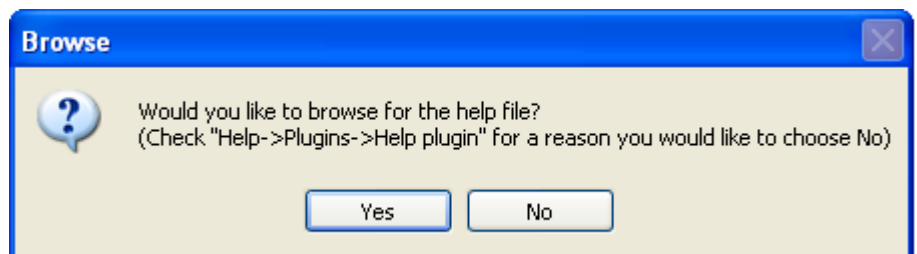
(1) 添加文件

用鼠标点击右上侧的Add按钮，得到对话框见右图。

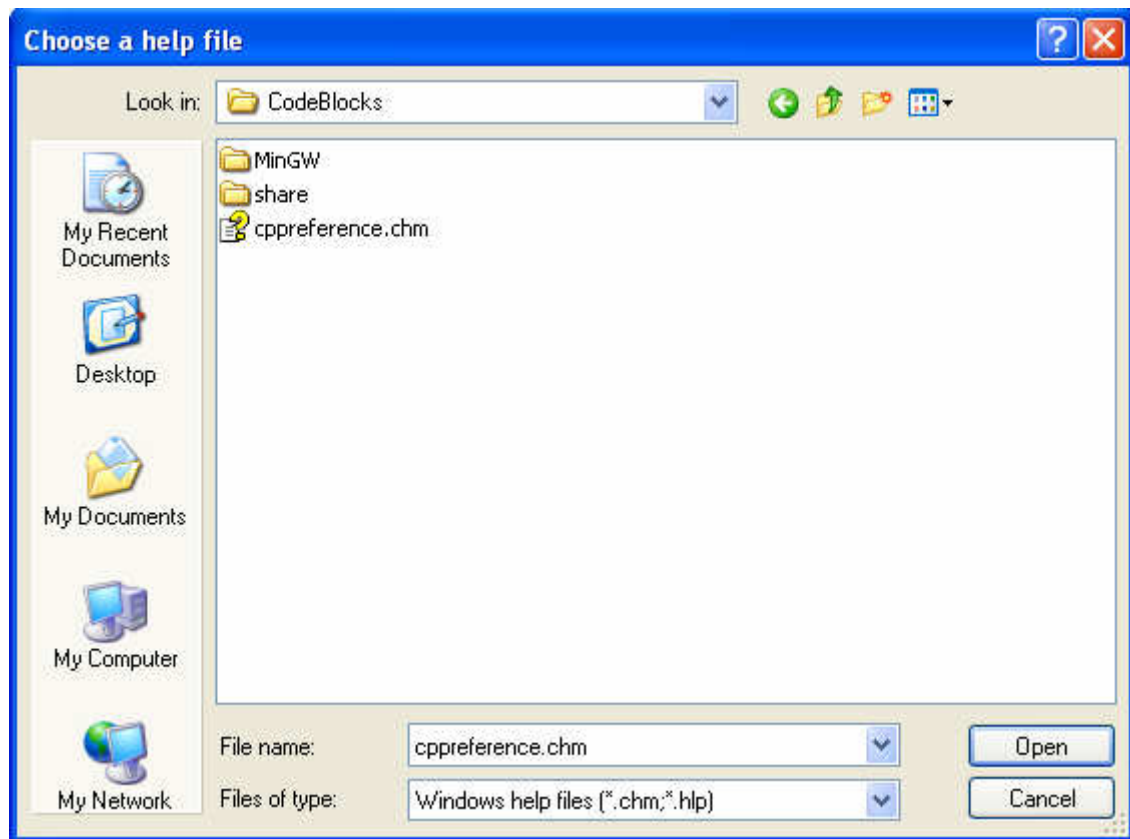


(2) 键入帮助文件题头

给添加的文件取一个题头名，该名字可以跟实际文件名相同，也可以不同，然后选择OK按钮，又弹出一个对话框见右图。

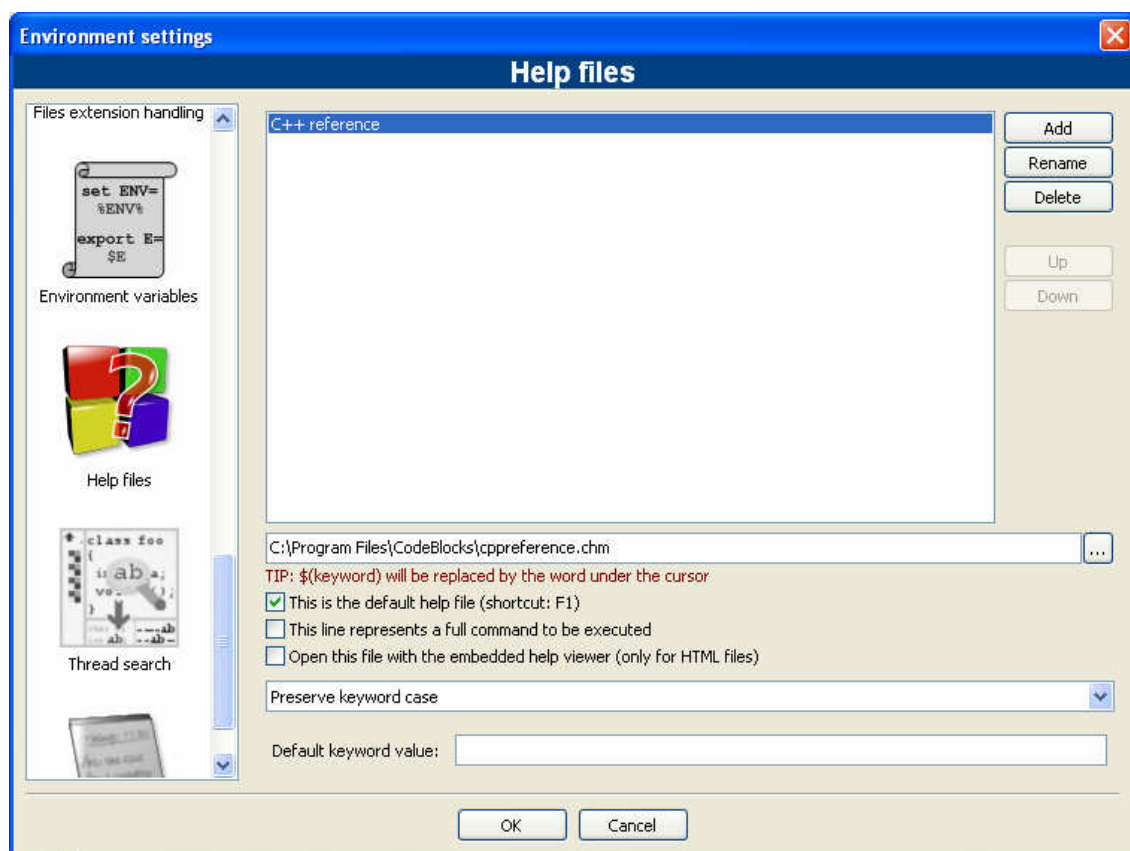


选择Yes按钮，进入下一步，见下图。



(3) 选择需要打开的文件

找到帮助文件的路径，选中帮助文件cppreference.chm，然后选择Open就又回到了刚进入Help files的对话框，只不过多了一行字C++ Reference，见下图。



并且有刚加载的文件cppreference.chm的对应路径。可以继续按照上述步骤添加更多帮助文件，也可以用右上侧的按钮Rename对题头C++ Reference进行改名或者用Delete按钮删除此题头。

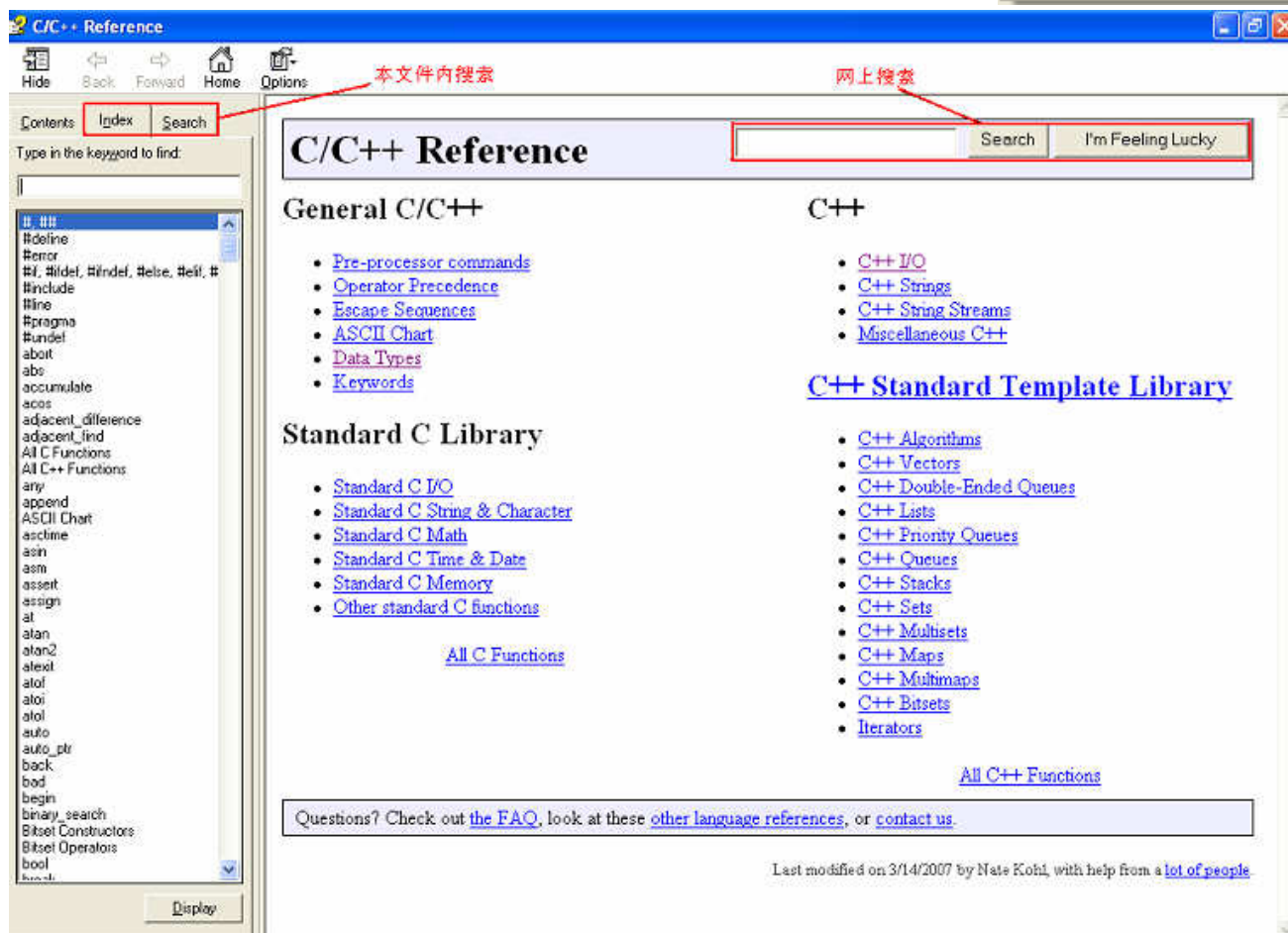
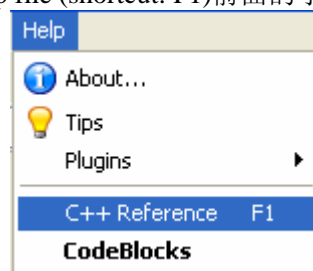
(4) 使帮助文件可用

为了方便使用，选中C++ Reference并用鼠标在下面的标签This is the default help file (shortcut: F1)前面的小方框中打勾，见上图，然后再用鼠标点击下面的OK按钮。

(5) 测试帮助文件是否可以成功加载

进入Code::Blocks(如果刚才您未退出Code::Blocks就无需再重新进入)，选择主菜单Help下的C++ Reference F1按钮，如右图。或者按下F1快捷键，就可以成功加载刚才设置需要加载的帮助文件cppreference.chm了。

经过上述这些设置后，Code::Blocks就可以成功加载帮助文件了，按下F1快捷键或菜单Help下的按钮C++ Reference F1会弹出这样的一个界面，见下图。



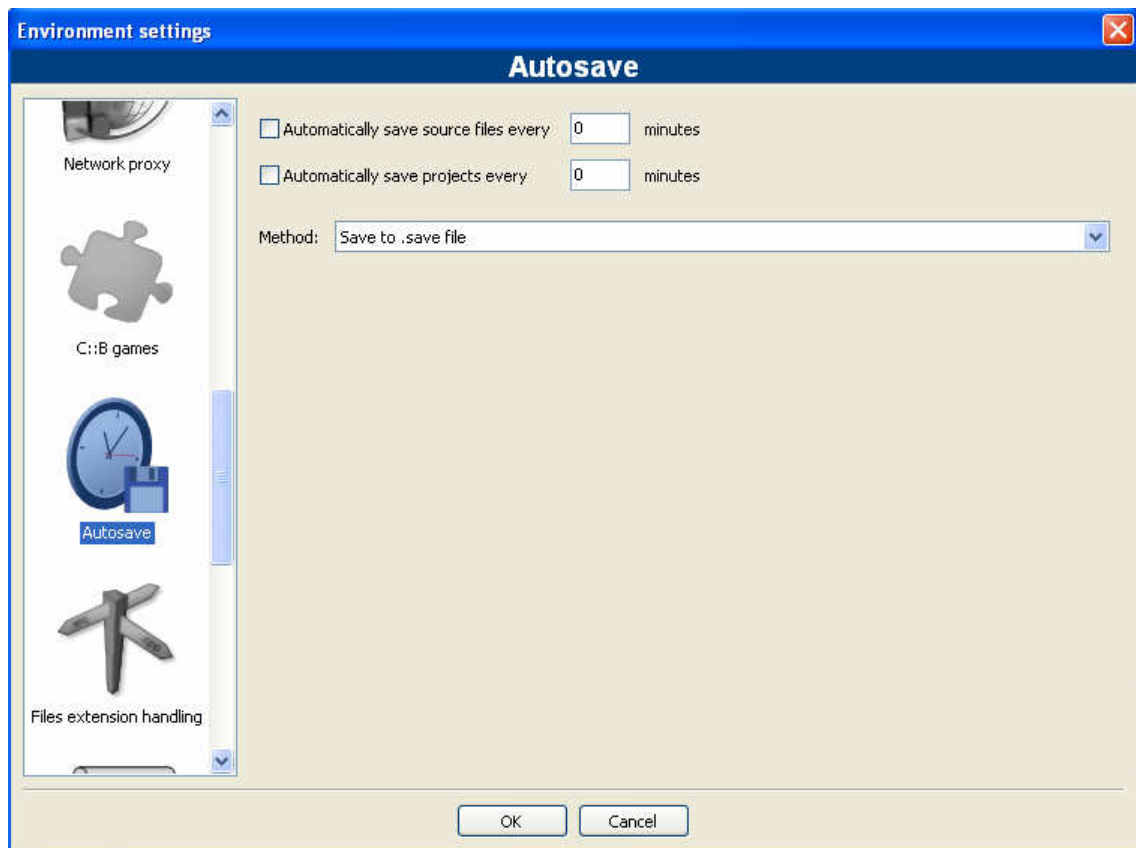
假如我们需要查阅标准的C++库函数，可以选择左侧的Index或Search按钮键入函数名进行查询。如果您使用的电脑已经联网，在左上部空框内键入要查询的函数名，再用鼠标点击右侧Search按钮就可以进行http://www.cppreference.com网上查询。

2.1.2 自动保存

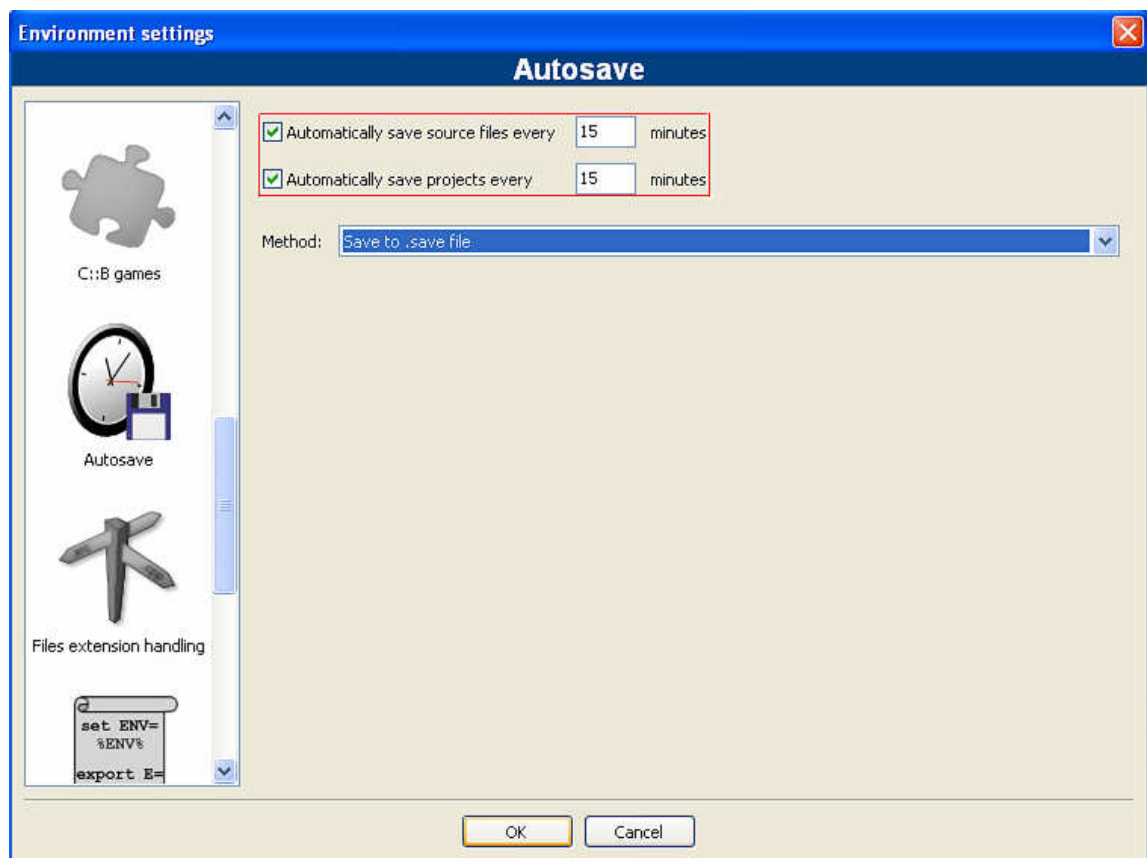
编写或者调试程序的过程中偶尔出现断电，如果没有后被电源，此时可能会丢失部分程序内容。为此，我们需要设置Code::Blocks能自动保存功能所对应的选项。

进入Code::Blocks后，选择主菜单Settings下Environment...子菜单，弹出一个对话框，用鼠标拖动左侧的滚动条，找到如右图标。选中它，界面如下图。





分别设置自动保存源文件和工程的时间，例如均为15分钟，见下图中红色框中部分。



Method为保存文件的方法，有三种，分别是Create backup and save to original file, Save to original file, 以及 Save to .save file。选择最后一个， Save to .save file就可以了，设置完毕后，选择OK按钮。

2.2 编辑器

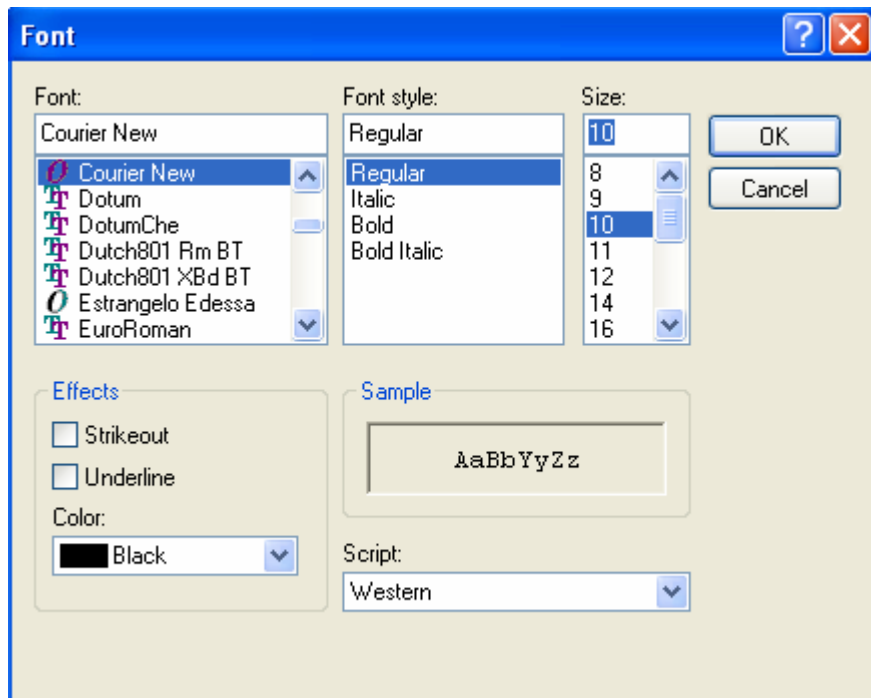
编辑器主要用来编辑程序的源代码，Code::Blocks内嵌的编辑器界面友好，功能比较完备，操作也很简单。

2.2.1 通用设置

启动Code::Blocks，选择主菜单Settings下的子菜单Editor...会弹出一个对话框，默认通用设置General settings栏目，选中一些选项如右下图。

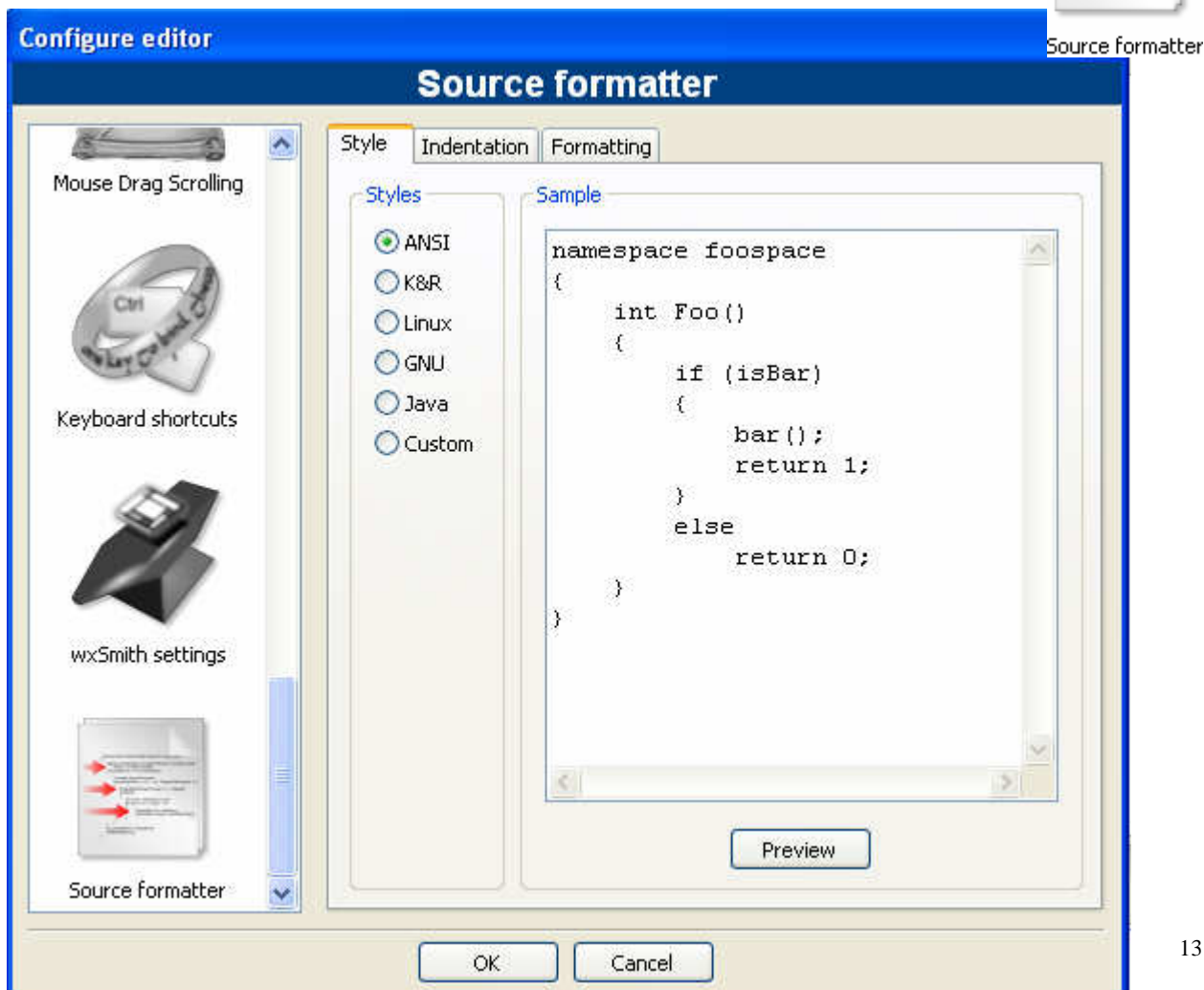
然后设置字体，字体设置首先选择右上角的Choose按钮，会弹出一个对话框，对话框主要有三个竖向栏目，最左侧的栏目Font:用来选择字体类型，选择Courier New，中间栏目Font style: 是字体样式，选择Regular，最右边的栏目Size:是文字大小，根据个人习惯和电脑显示器显示面积大小进行选择，一般10~12，其它选项不变。见右图。

然后用鼠标选择OK，则字体参数设置完毕，进入上一级对话框General settings，再选择OK，则General settings设置完毕，回到Code::Blocks主界面。



2.2.2 源代码格式

不同的人编写代码风格不同，Code::Blocks提供了几种代码的书写格式。首先从Settings主菜单进入子菜单Editor...，然后从弹出的对话框中移动滚动条，找到标签为Source formatter的按钮，见右图。选中它，可以看到右侧Style菜单下有几种风格分别为ANSI, K&R, Linux, GNU, Java, Custom, 最右侧则是这些风格的代码预览Preview。可以根据个人习惯进行选择，如果选择Custom则需要自己设置两个子菜单Indentation和Formatting下的各个选项，选中自己习惯或者喜欢的风格(笔者的习惯是用ANSI)，然后点击OK按钮。



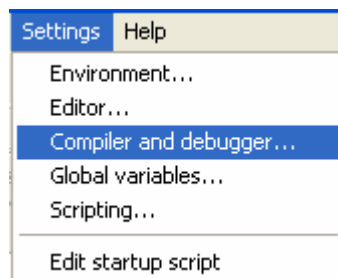
如此以来，编辑器的基本设置就完成了，尽管还有很多其它的选项和参数，但是并不太常用，因此这里就不做详细介绍了。

2.3 编译器和调试器

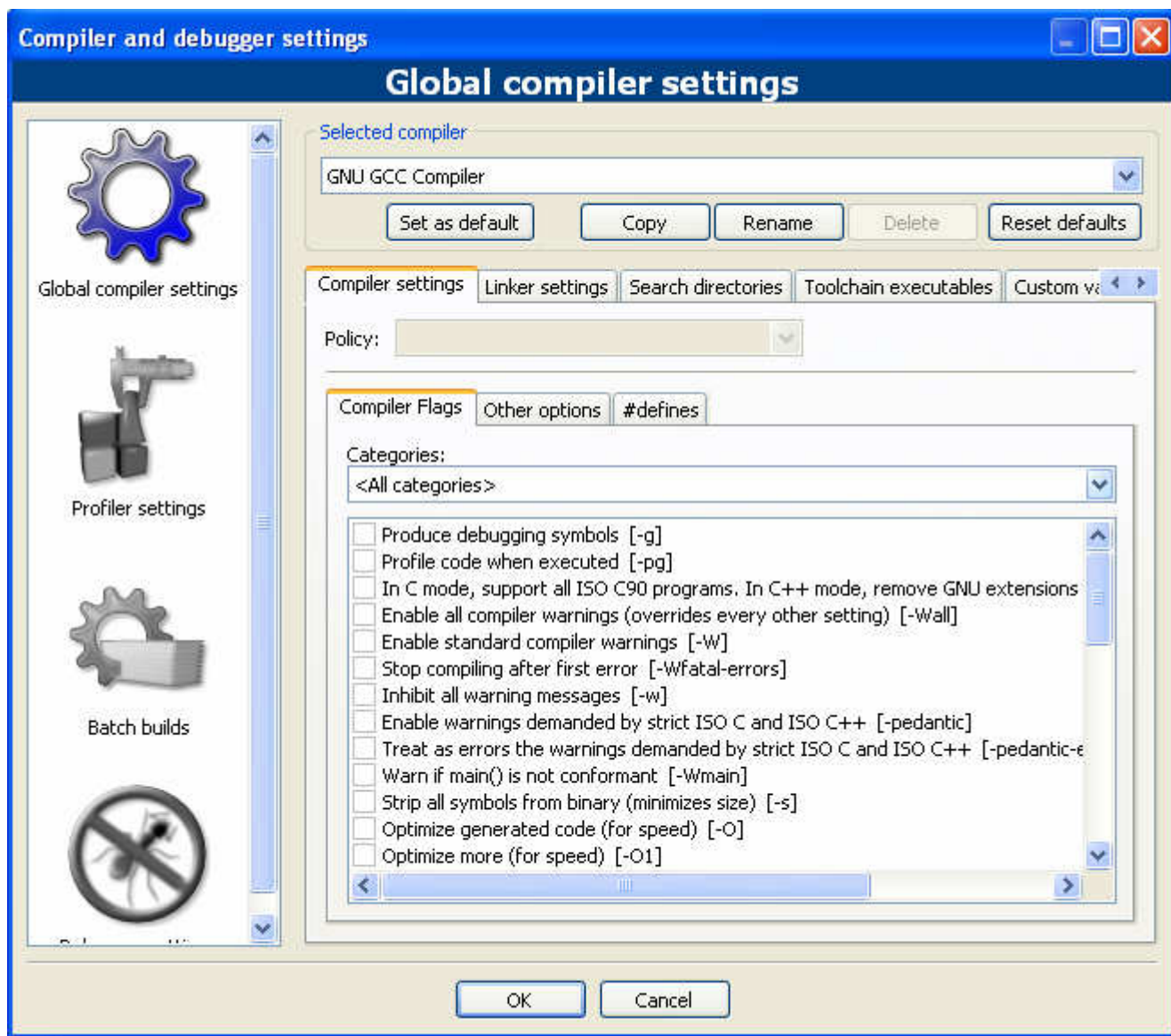
编译器和调试器可能最重要，因为编写的源代码需要转换成机器可以识别的二进制才能执行，而且编写程序时，很难保证一次正确。这里主要讲述一下，编译器和调试器的全局配置，也就是说这里配置的每个选项都会影响到将来建立的每个工程。

2.3.1 进入编译器和调试器配置界面

首先从Code::Blocks界面的主菜单Settings下拉菜单下选择Compiler and debugger...按钮，见右图。

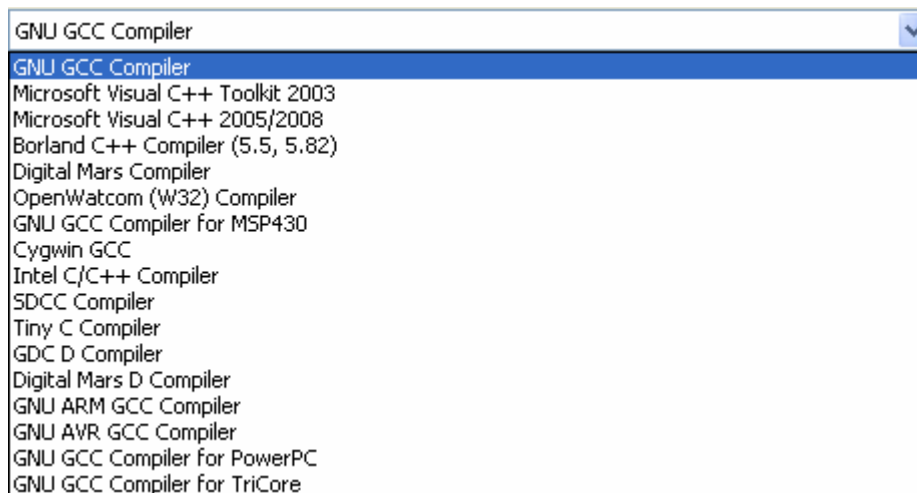


会弹出一个对话框如下，这就是编译器和调试器的配置界面，见下图。



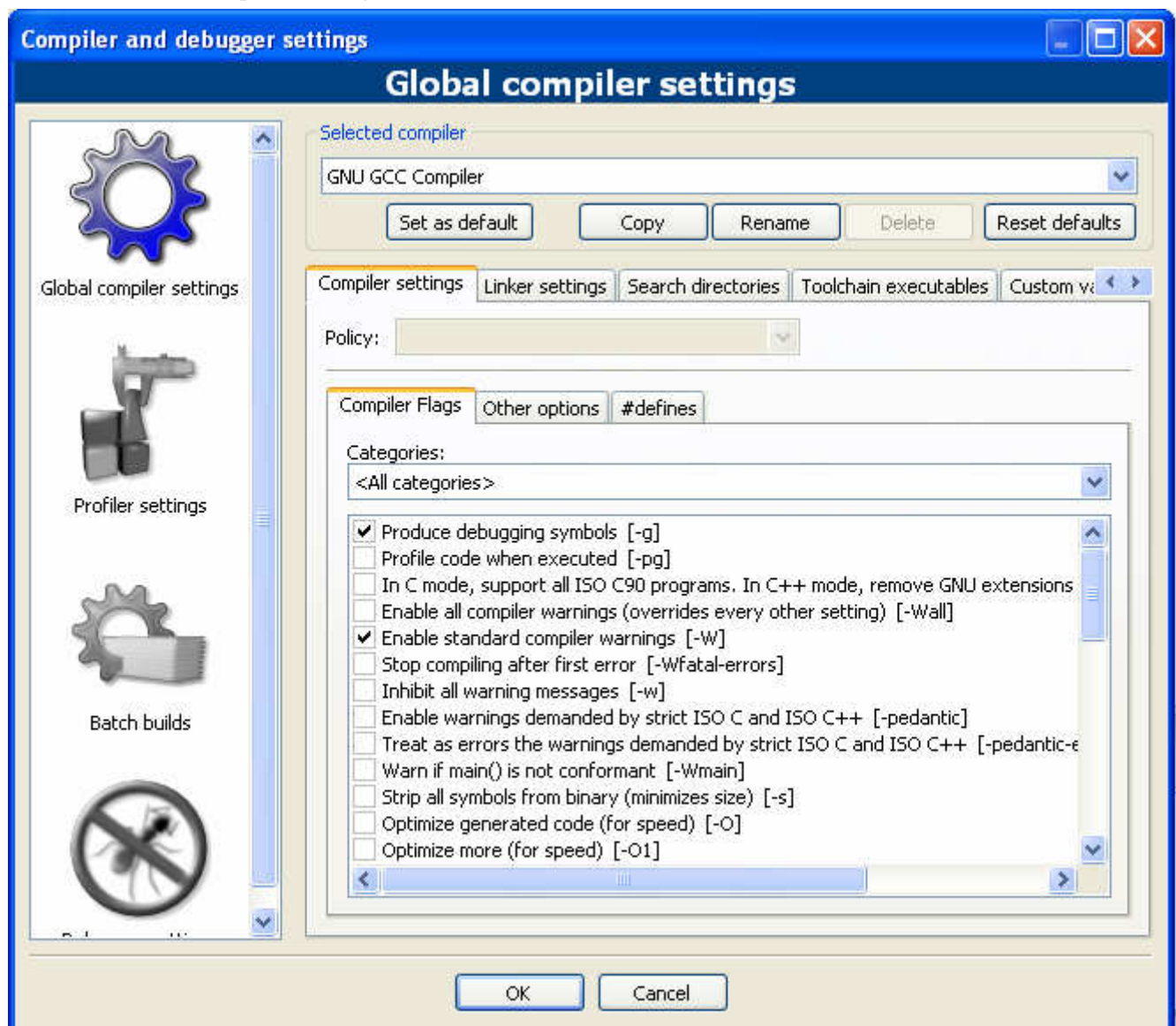
2.3.2 编译器选择

Code::Blocks支持多种编译器，默认编译器GNU GCC Compiler，当然，您也可以选择其它的编译器，只不过需要事先安装好您想用的编译器，见下图。

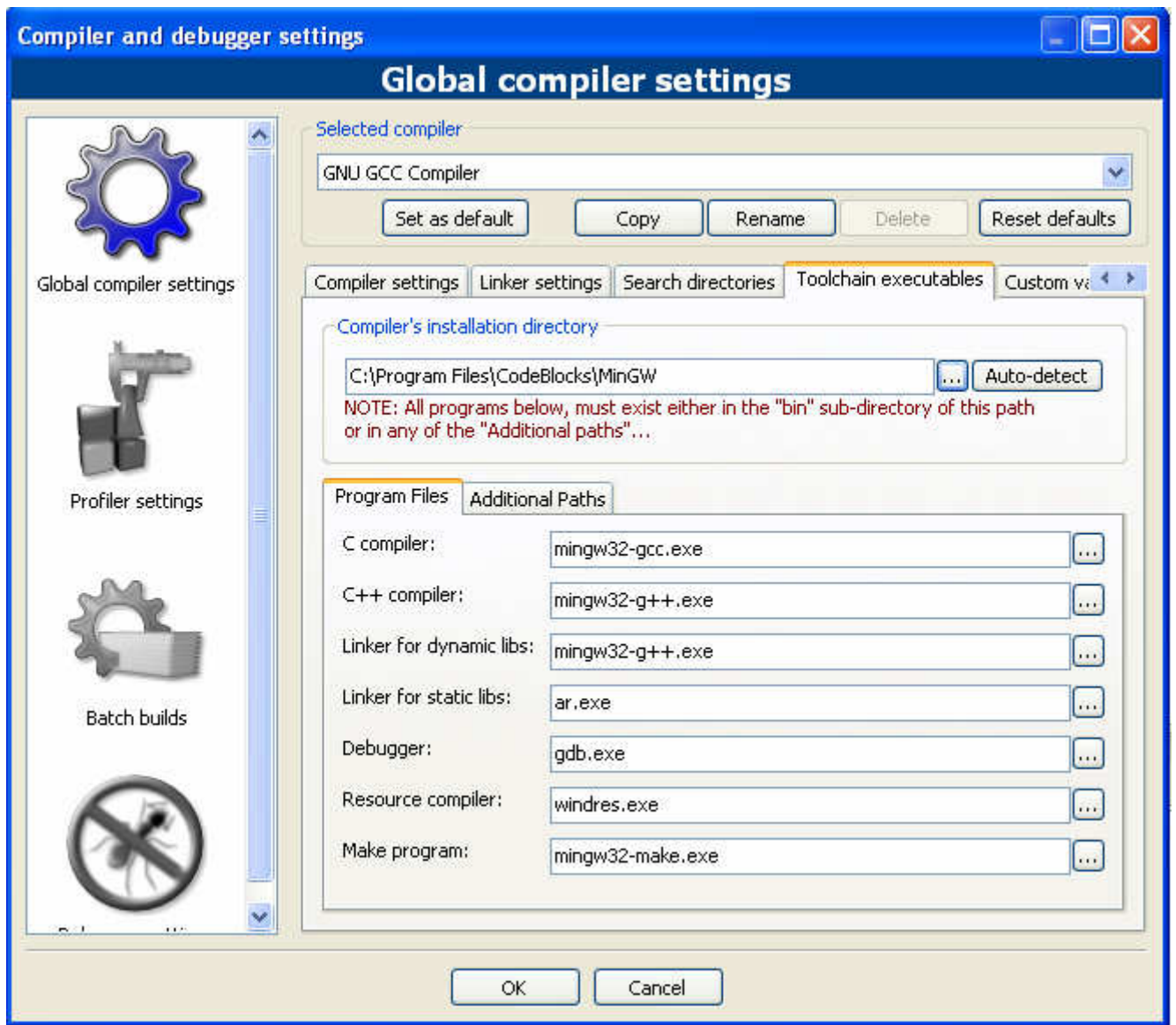


2.3.3 扩展编译选项配置

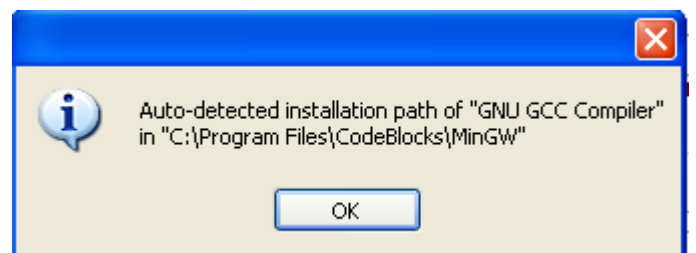
选择Compiler Settings菜单下的Compiler Flags子菜单，选中其中两个选项，Produce debugging symbols [-g] 和Enable standard compiler warnings [-W]，也可以什么都不选，见下图。




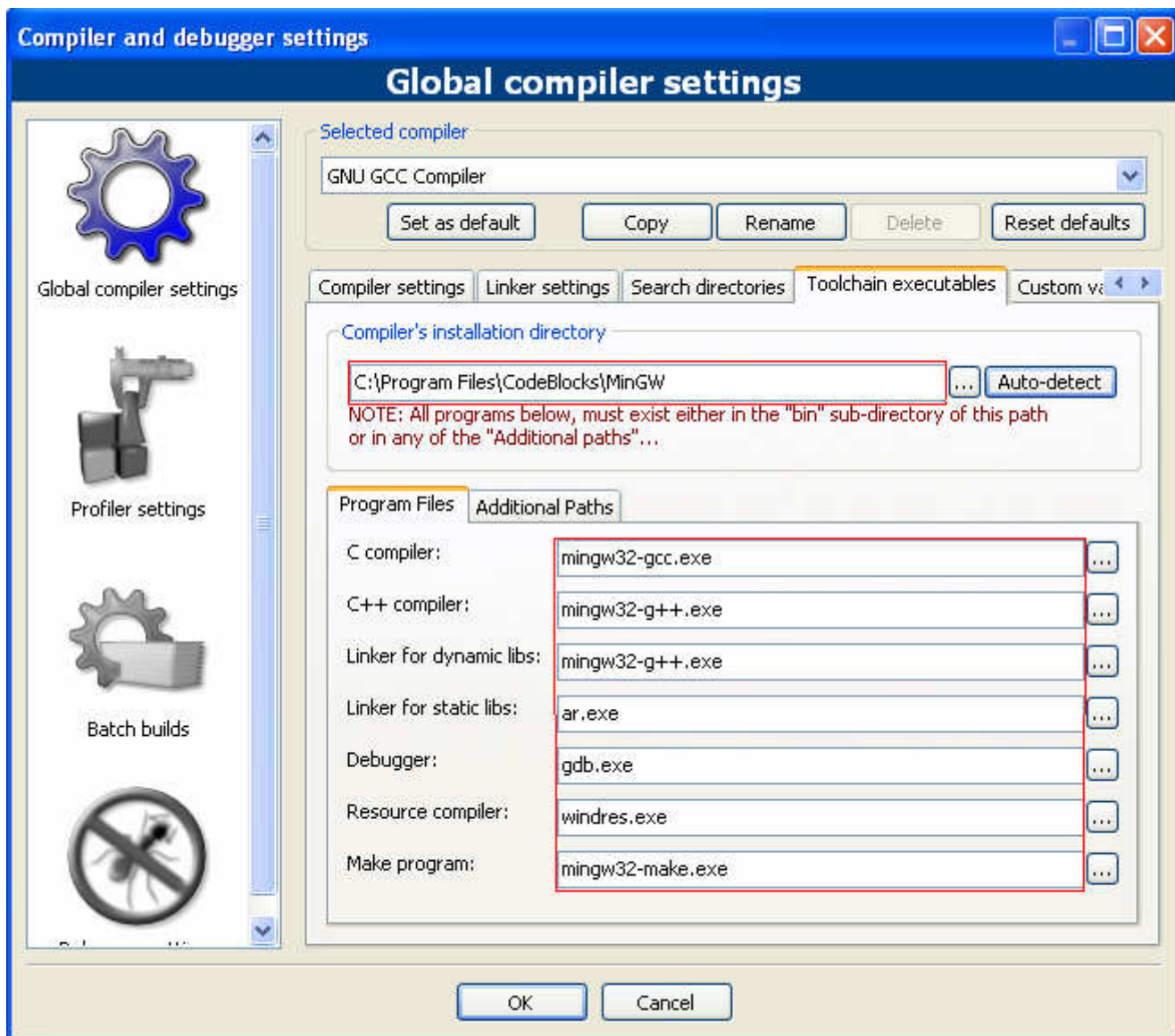
然后选择Toolchain executables子菜单，会出现一个界面，见下图。



点击右侧的Auto-detect按钮，一般而言能自动识别编译器的安装路径，见右图。



如果不能自动识别编译器安装的路径，就需要点击按钮进行手工配置好该路径。并且也要配置好C compiler:, C++ compiler:, Linker for dynamic libs:, Linker for static libs:, Debugger:, Resource compiler:, Make program:这几个选项的文件名。见见下图中用红色方框框起来的部分。



最后用鼠标点击最下方的OK按钮，则编译器和调试器基本配置完毕。