Fundamental of C Programming

by

Dr. Md. Sherajul Islam

Professor

Department of Electrical and Electronic Engineering

Khulna University of Engineering & Technology

Outlines

- *Why C Language*
  ???

Why C Language ???

- **Most Commonly used P.L. for OS**
  - **Unix was first OS**
  - **Later MS Windows**
  - **Linux**
- **Inspiration for other popular high level language available today**
  - **Perl, PHP, Python etc...**

Knowing C Will enable us to understand and appreciate an entire family of programming languages built upon the tradition of C

Classification of Language

PL are mainly classified into three categories

- **Low Level Language**
- **Medium Level Language**
- **High Level Language**

Fortran, C, Java are high level language

C Compiler

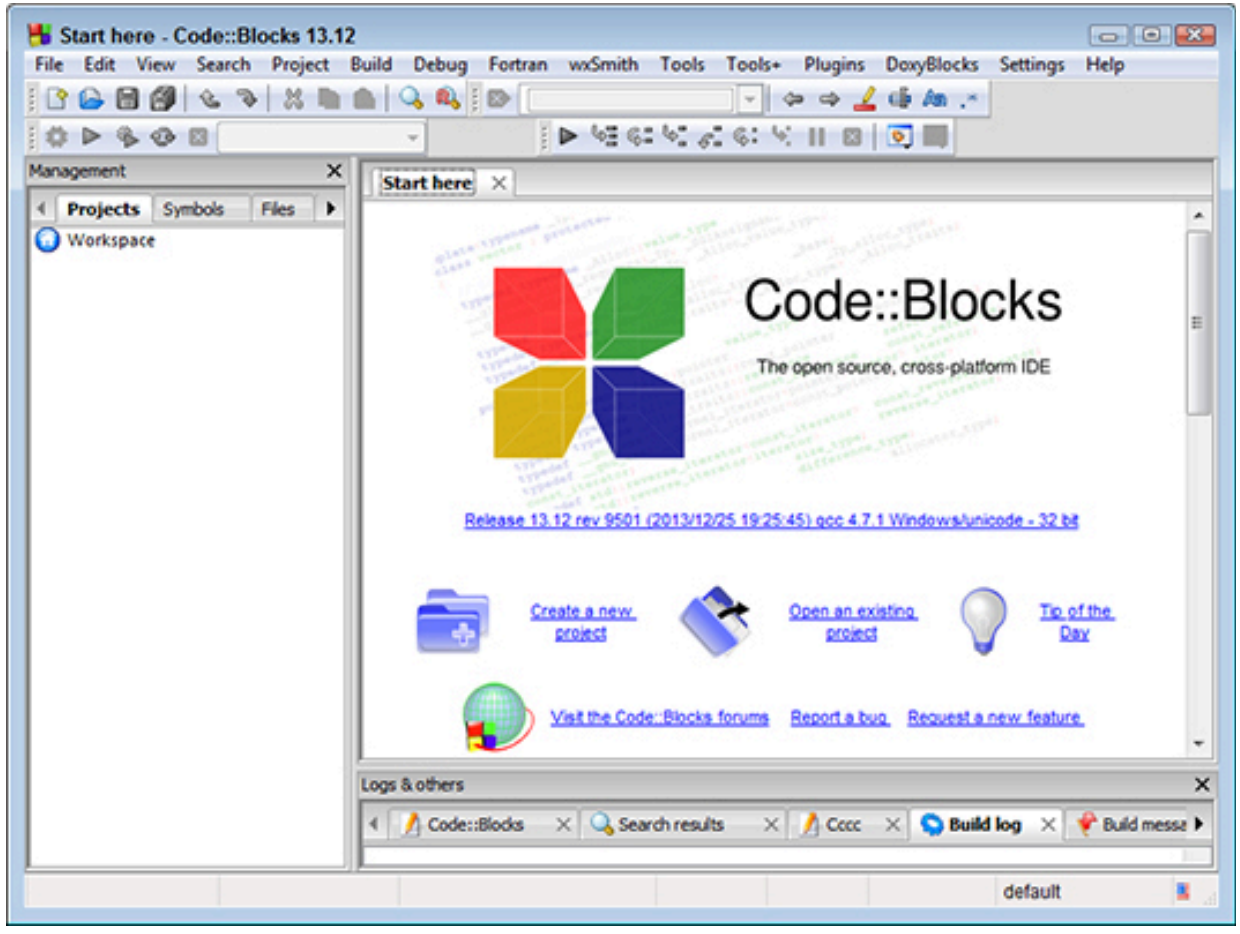
Compiler converts the C code into executable machine code

- **MS Visual Studio Express**
- **Tiny C Compiler (tcc)**
- **GNU C Compiler (gcc)**

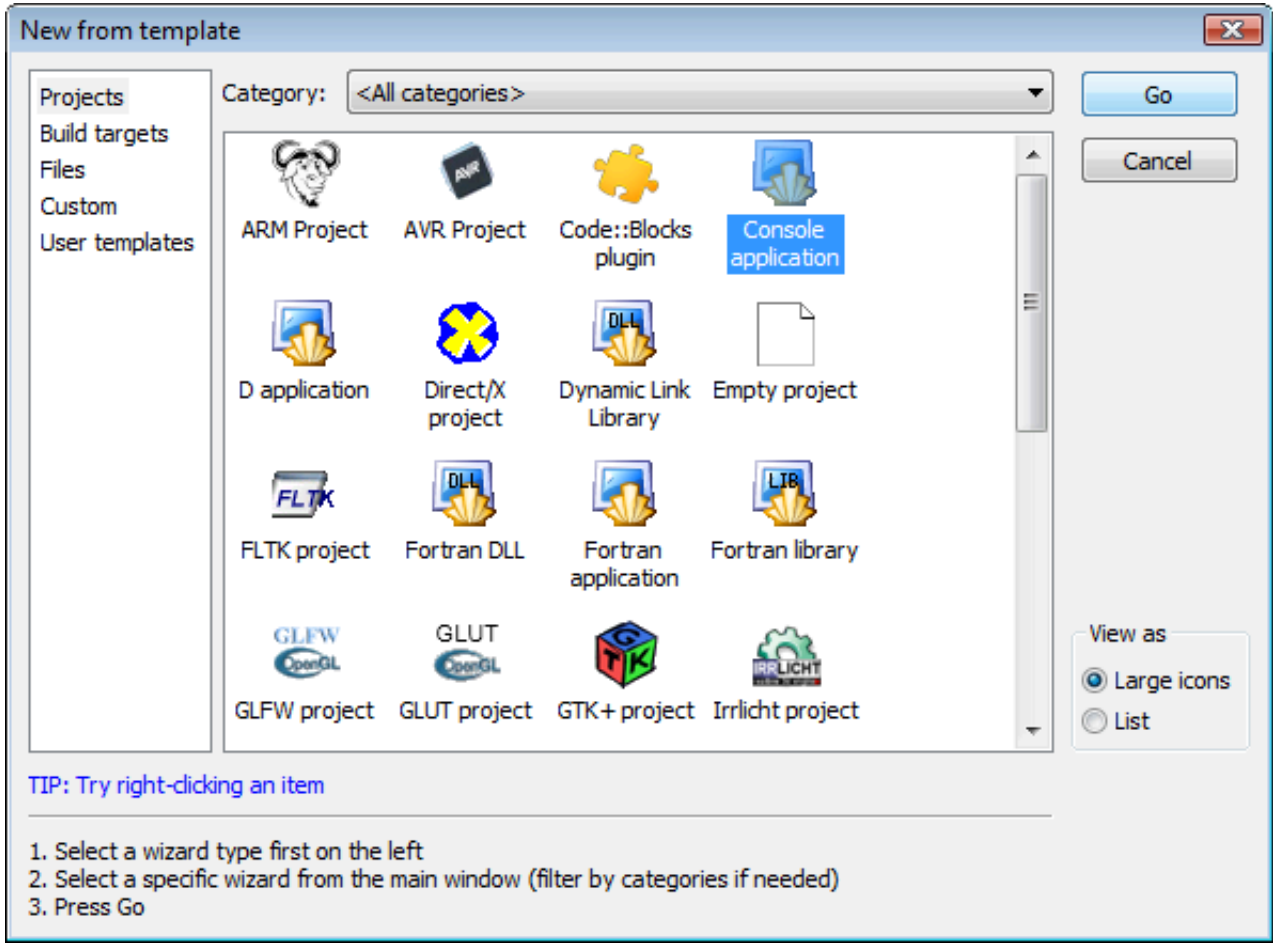
Minimum Software requirements to program in C is a text editor

C Compiler

***Code::Blocks** is a free integrated development environment (IDE) for **C** and **C++** on Windows, Linux and MacOS X.*

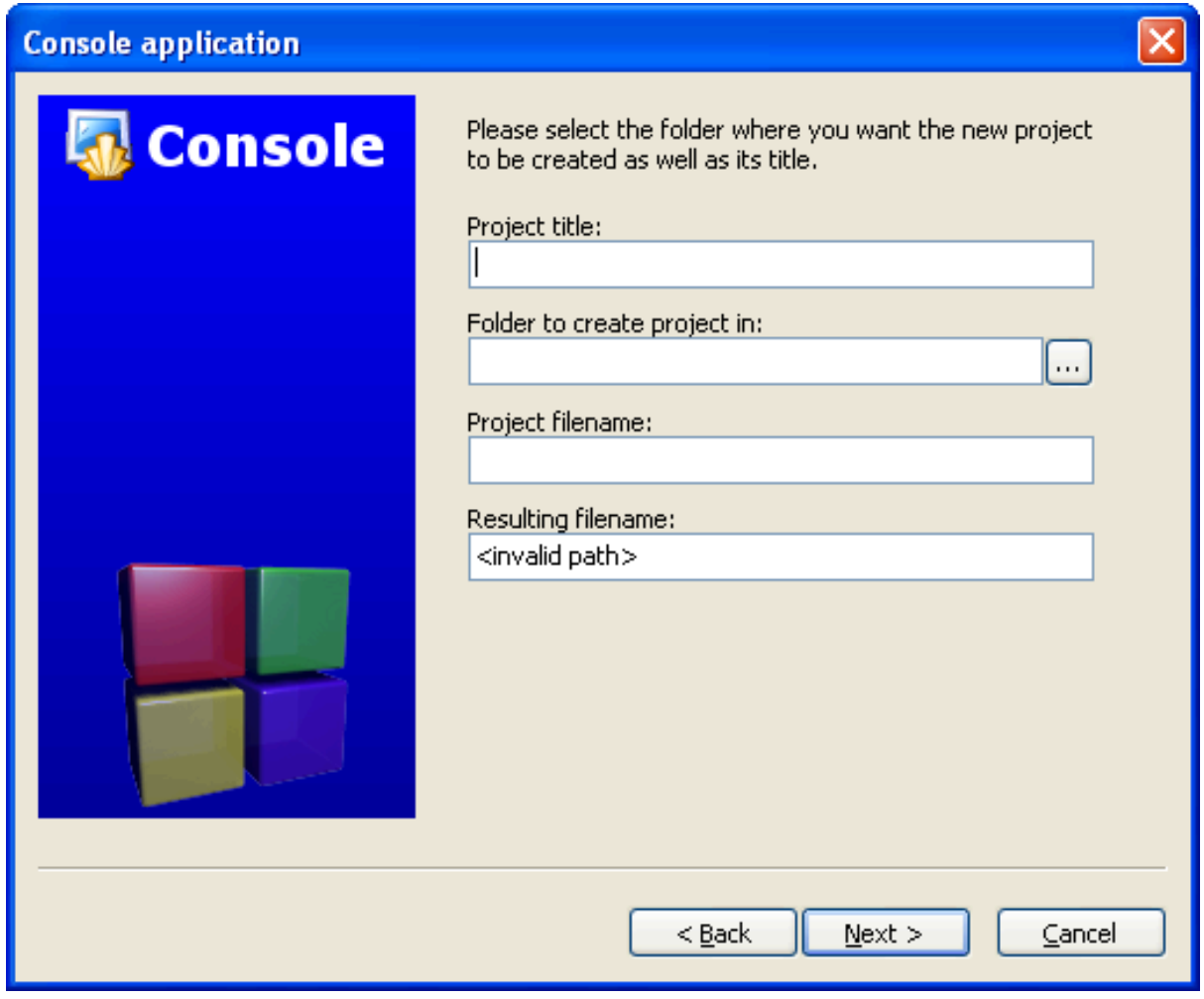


C Compiler

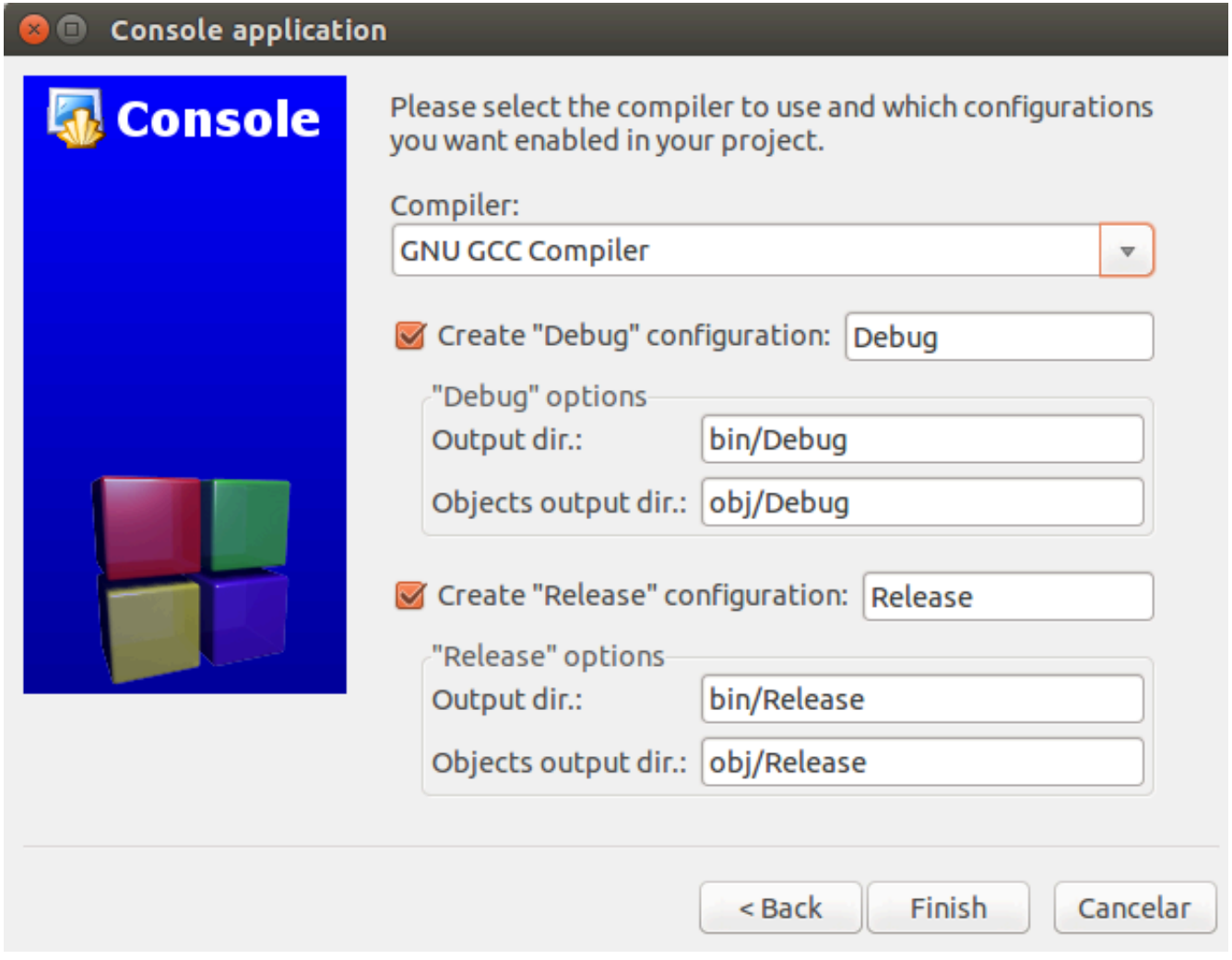

New from template
Projects
Build targets
Files
Custom
User templates
Category:
<All categories>
Go
Cancel
ARM Project
AVR Project
Code::Blocks plugin
Console application
D application
Direct/X project
Dynamic Link Library
Empty project
FLTK project
Fortran DLL
Fortran application
Fortran library
GLFW project
GLUT project
GTK+ project
Irrlicht project
View as
Large icons
List
TIP: Try right-clicking an item
1. Select a wizard type first on the left
2. Select a specific wizard from the main window (filter by categories if needed)
3. Press Go

C Compiler

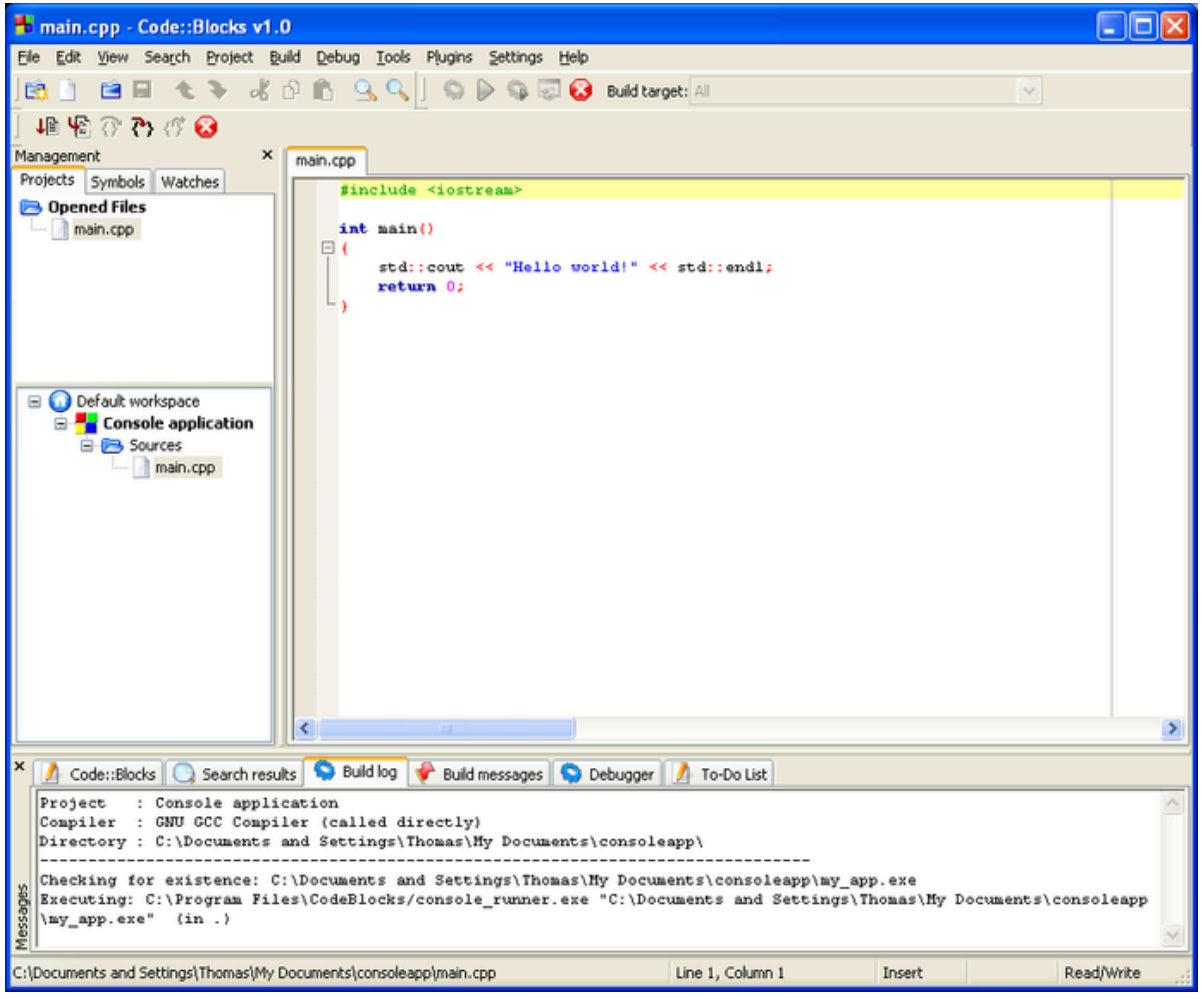
C Compiler


Console application
Console
Please select the folder where you want the new project to be created as well as its title.
Project title:
Folder to create project in:
...
Project filename:
Resulting filename:
<invalid path>
< Back
Next >
Cancel

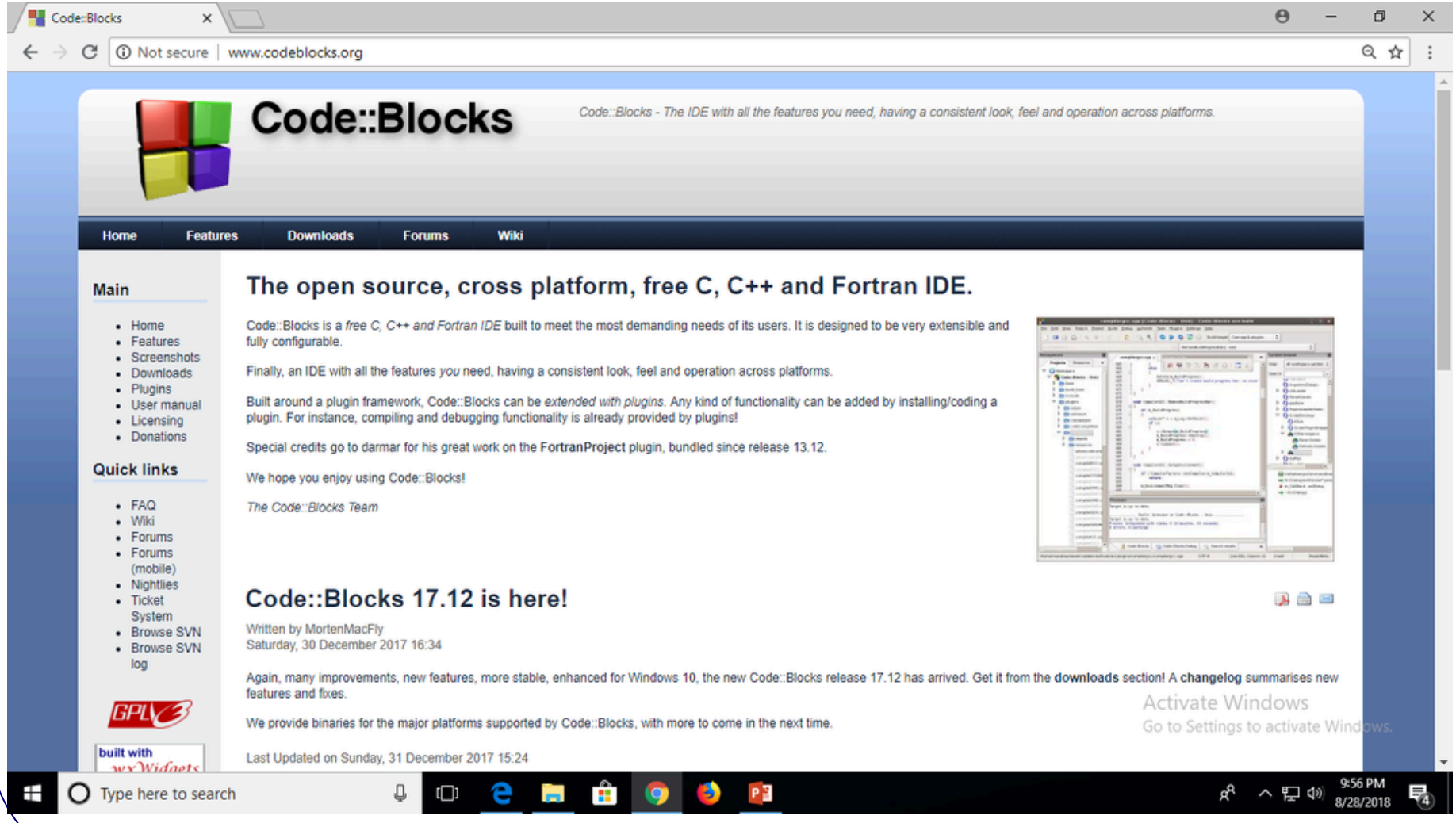
C Compiler


Console application
Console
Please select the compiler to use and which configurations you want enabled in your project.
Compiler:
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
< Back
Finish
Cancelar

C Compiler


main.cpp - Code::Blocks v1.0
File Edit View Search Project Build Debug Tools Plugins Settings Help
Build target: All
Management
Projects Symbols Watches
Opened Files
main.cpp
Default workspace
Console application
Sources
main.cpp
main.cpp
#include <iostream>
int main()
{
std::cout << "Hello world!" << std::endl;
return 0;
}
Code::Blocks Search results Build log Build messages Debugger To-Do List
Project : Console application
Compiler : GNU GCC Compiler (called directly)
Directory : C:\Documents and Settings\Thomas\My Documents\consoleapp\
Checking for existence: C:\Documents and Settings\Thomas\My Documents\consoleapp\my_app.exe
Executing: C:\Program Files\CodeBlocks/console_runner.exe "C:\Documents and Settings\Thomas\My Documents\consoleapp
\my_app.exe" (in .)
Messages
C:\Documents and Settings\Thomas\My Documents\consoleapp\main.cpp
Line 1, Column 1
Insert
Read/Write

How to download Code::Blocks



Origin of C Language

- **First in Bell Laboratories**
  - **Dennis Ritchie**
- **BCPL or B language**

B language was modified into C language

General Structure of C Code

- **Basic Structure of C**
  - **Documentation Section**
  - **Link Section**
  - **Definition Section**
  - **Global Declaration Section**
  - **Main Function Section**
  - **Declaration Section**
  - **Executable Part**

```c
// Documentation Section: Program to add two numbers
// Author: [Your Name]
// Date: September 2, 2025

// Link Section: Include standard input/output library
#include <stdio.h>

// Definition Section: Define a constant
#define MAX_VALUE 100

// Global Declaration Section: Declare a global variable
int globalVar = 10;

// Function prototype declaration
int addNumbers(int a, int b);

// Main Function
int main() {
    int num1, num2, sum;

    printf("Enter two numbers: ");
    scanf("%d %d", &num1, &num2);

    sum = addNumbers(num1, num2); // Call user-defined function

    printf("Sum = %d\n", sum);
    printf("Global Variable: %d\n", globalVar);

    return 0; // Indicate successful execution
}

// Subprogram (User-Defined Function)
int addNumbers(int a, int b) {
    return a + b;
}
```