

# INSTALLAZIONE E UTILIZZO DEL COMPILATORE Code::Blocks 8.02

## Download

Si può scaricare gratuitamente la versione per il proprio sistema operativo (Windows, MacOS, Linux) dal sito: <http://www.codeblocks.org> alla voce Download. A meno che non si voglia usare un compilatore già installato sul proprio sistema, si scarichi la versione con le librerie *MinGW*, il file di installazione ha nome: `codeblocks-8.02mingw-setup.exe`.

## Installazione (Windows)

Doppio click sul file scaricato, quindi:

1. Next
2. I agree per accettare le condizioni di utilizzo
3. Nella sezione **Choose components** verificare che l'ultima voce ("*MinGW Compiler Suite*") sia selezionata quindi Next
4. Indicare la directory (cartella) dove installare **Code::Blocks**, il default per Windows è `C:\Program Files\CodeBlocks`, quindi premere Install
5. No alla richiesta di lanciare Code::Blocks
6. Next dopo che i file sono stati installati
7. Finish

## Creazione di un progetto

Far partire il programma.

Qualora ci fosse una schermata per la scelta del compilatore, selezionare il GCC che dovrebbe avere l'indicazione "*detected*" (cioè trovato sul sistema), premere il bottone sulla destra **Set as default** e quindi continuare, alla successiva richiesta di associare i file ".c" a **Code::Blocks** lasciare la scelta corrente e procedere.

Un programma può essere composto da uno o più file sorgenti che dopo la compilazione vengono assemblati dal linker per costituire un unico programma eseguibile, questi sorgenti vengono collettivamente identificati come un unico **progetto**. Per creare un progetto, lanciato il programma, si seguano i passi indicati:

1. Al centro della schermata iniziale fare click su "*Create a new project*", oppure utilizzare il menu **File/New/Project**



[Create a new project](#)



[Open an existing project](#)



[Visit the Code::Blocks forums](#)

[Report a bug](#)

[Request a new feature](#)



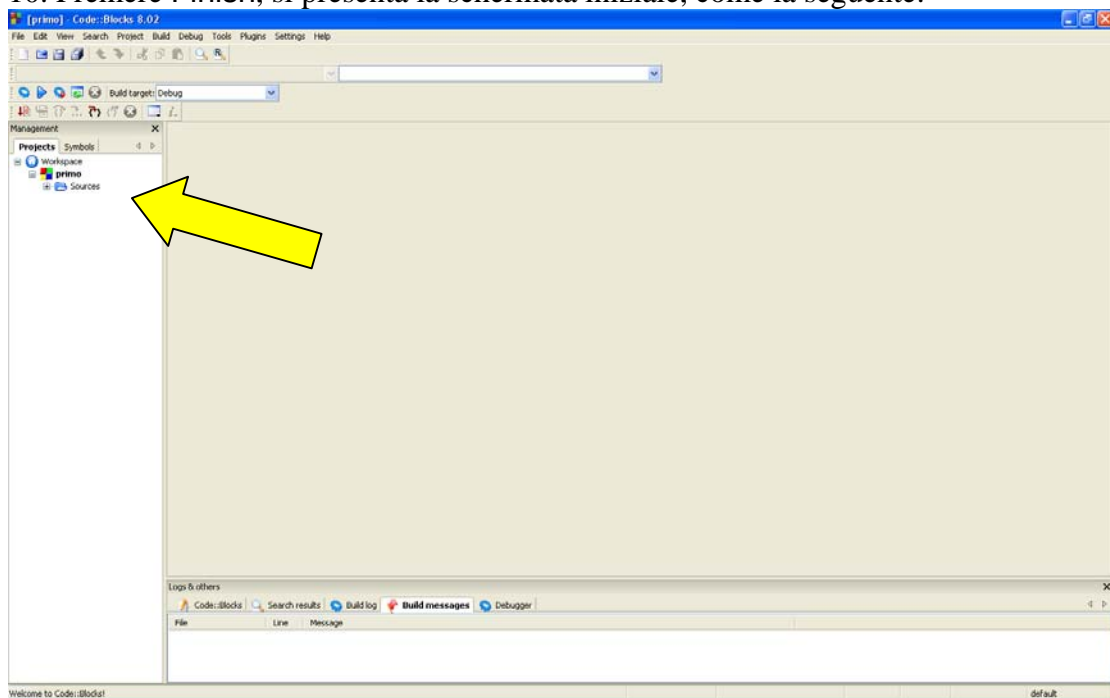
### Recent projects

- [E:\Program Files\CodeBlocks\Projects\primo\primo.cbproj](#)

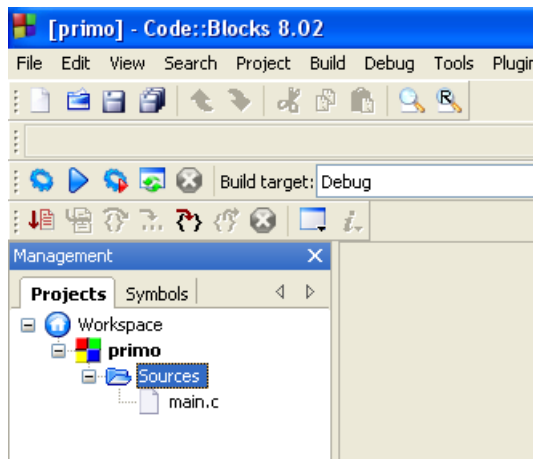
### Recent files

No recent files

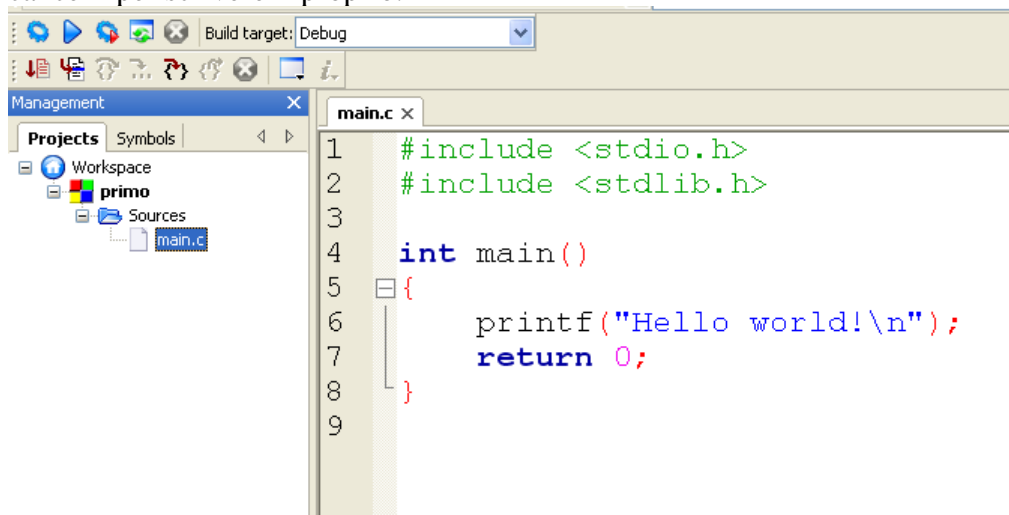
2. Nella finestra “*New from template*” selezionare “*Console Application*” e premere il bottone **Go**
3. Premere **Next**
4. Selezionare il compilatore **C** e **NON C++** e premere **Next**
5. Dare un nome al progetto in “*Project Title*”, ad esempio “*esercizio4.2*”
6. Se il campo “*Folder to create object in.*” è vuoto, aggiungere la directory dove salvare i file relativi al progetto, ad es. “*C:\Programmi\CodeBlocks\Projects\*”
7. I restanti campi vengono modificati automaticamente, non cambiarli
8. Premere **Next**
9. Selezionare il compilatore: “*GNU GCC compiler*” se non è selezionato, non modificare gli altri campi (segni di spunta nei due quadrati, *checkbox*)
10. Premere **Finish**, si presenta la schermata iniziale, come la seguente:



11. Modificare le impostazioni se desiderato:
  - a. per cambiare il numero di caratteri di indentazione, si veda la voce di menu: **Settings/Editor**, si consiglia di impostare 3 o 4
  - b. per cambiare la dimensione del testo, si selezioni la voce di menu: **Settings/Editor** il bottone **Choose**
12. Modificare le impostazioni:
  - a. per poter vedere il valore delle variabili semplicemente posizionando sopra di esse il mouse, identificare dal menu **Settings/Compiler and debugger** l'elemento verticale a sinistra con la dicitura **Debugger settings** e selezionare la voce “*Evaluate expression under cursor*” e chiudere premendo **OK** in questa finestra e la successiva.
13. Nella schermata iniziale, nel riquadro di sinistra in alto (vedere la freccia nella figura), si può identificare il nome del progetto in grassetto (qui è **primo**) avente al di sotto la cartella “*Sources*”, aprendo questa con un doppio click si vede un file denominato `main.c`, si cambi il nome se si desidera **ma non si usino nomi contenenti spazi, inoltre si mantenga l'estensione “.c”**



14. Si faccia ora un doppio click sul nome del file “.c”, si apre una schermata con alcune righe standard che costituiscono un programma C minimale, lo si cancelli per scrivere il proprio:



15. Salvare il file con File/Save as, i salvataggi successivi si possono ottenere con un semplice Control-S

### **Apertura di un progetto già creato**

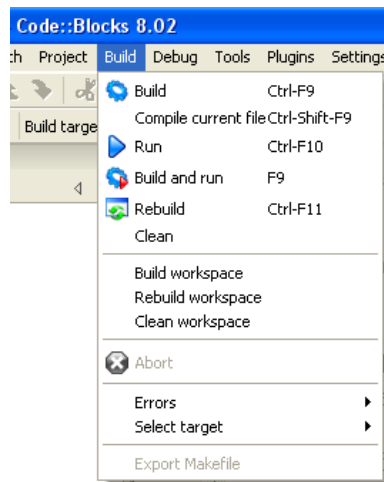
E' possibile aprire un progetto già creato facendo click sulla voce “Open an existing project...” che si ottiene quando **Code::Blocks** viene mandato in esecuzione.,oppure direttamente sul suo nome in “Recent projects” (vedere figura al punto 1 della sezione precedente a questa). Le stesse opzioni si trovano nel menu File.

### **Compilazione ed esecuzione**

Questi comandi si trovano sia in una delle toolbar:



sia nel menu Build:



Ci sono le stesse icone, ad es. si vede che il triangolo verde equivale a **Control-F10**, posizionando il mouse sulle icone della toolbar appare l'indicazione di che cosa siano. Quando la compilazione viene effettuata vengono segnalati gli errori sintattici e i warning e il numero della riga dove questi si sono verificati. La zona grigia a fianco delle righe di codice riporta i numeri delle righe. Un doppio click sull'indicazione dell'errore posiziona il cursore alla riga con l'errore e un quadratino rosso la identifica. Non necessariamente l'errore è alla riga indicata (ad es. la mancanza di un punto e virgola viene rilevata solo alla riga successiva), inoltre un errore può generarne molti altri in cascata, questi in genere vengono eliminati dalla correzione dell'errore iniziale.

**Control-Shift-F9** – (non è nella toolbar)

compila il file corrente (non fa il link), se è stato modificato

**Control-F9 - Build**

compila e fa il link (fa il Build), se è stato modificato

**Control-F10 - Run**

esegue il programma (non si ferma ai breakpoint)

**F9 – Build and run**

fa il build ed esegue il programma (non si ferma ai breakpoint)

**Control-F11 - Rebuild**

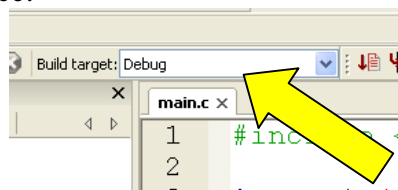
compila e fa il link (fa il Build), anche se non è stato modificato

## ***Argomenti sulla riga di comando***

Per indicare gli argomenti sulla riga di comando, selezionare nella barra dei menu: Projects/Set program's arguments...

## ***Come fare il debug di un programma***

Si hanno a disposizione diversi strumenti per fare il debug di un programma, questo deve essere stato compilato nella modalità *Debug* (è il default), si può vedere se è così nella schermata con il codice:



Quando il programma funziona correttamente è possibile selezionare la modalità *Release* in cui il compilatore ottimizza il codice modificandolo (ad esempio per avere

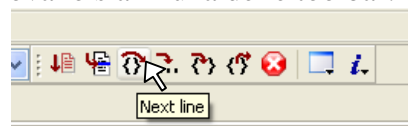
la massima velocità di esecuzione), il codice ottimizzato è diverso e di più difficile comprensione rispetto al codice di partenza.

## **Ricerca errori**

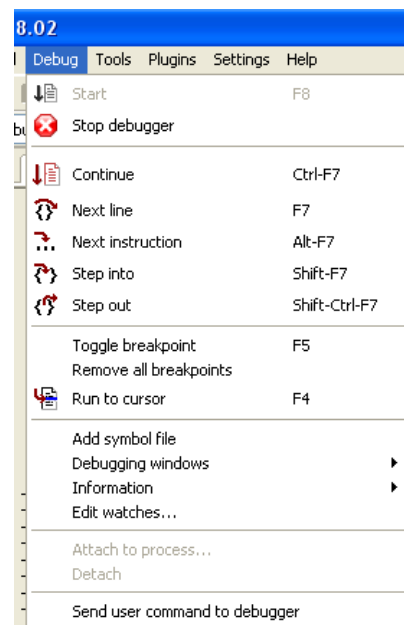
Quando il programma viene compilato correttamente non ha errori sintattici, ma non necessariamente produce i risultati voluti, questo perché il programmatore ha sbagliato a realizzare l'algoritmo (errori logici).

La ricerca degli errori logici di un programma viene chiamato *debug* e il metodo più efficace è quello di eseguire il programma istruzione per istruzione (passo passo): dopo che ogni istruzione viene eseguita il programma si ferma per permettere al programmatore di osservare (*watch*) il contenuto delle variabili: si vede quali istruzioni vengono eseguite e il cambiamento che queste attuano sulle variabili. La riga che verrà eseguita è preceduta da un triangolino giallo.

I comandi per il debug si trovano sia in una delle toolbar:



sia nel menu Debug:



Ci sono le stesse icone, posizionando il mouse sulle icone della toolbar appare l'indicazione di che cosa siano.

### **F8 - Start**

esegue il programma e si ferma ai *breakpoint*; se necessario prima compila e fa il link

### **Control-F7 - Continue**

continua l'esecuzione se precedentemente interrotta

### **Alt-F7 – Next instruction**

esegue una sola istruzione (per volta) del programma (modalità passo passo), se è una funzione, la esegue senza fermarsi

### **F7 – Next line**

esegue una sola riga (per volta) del programma (modalità passo passo), in genere un programma ha una sola istruzione per riga per cui equivale al precedente (ma è più comodo da utilizzare)

### Shift-F7 – Step into

Se l'istruzione da eseguire è una funzione, entra ad eseguire passo passo le istruzioni (Next instruction e Next line la eseguono come se fosse un'unica istruzione, senza eseguire passo passo le istruzioni)

### Debug/Stop Debugger

ferma definitivamente il programma (se eseguito passo passo)

Per far partire un programma in modalità passo passo occorre posizionare un breakpoint alla riga dove inizia il codice operativo (dopo le definizioni delle variabili) e far partire il programma in modalità debug (**F8**), NON con **Run (Control-F10)**

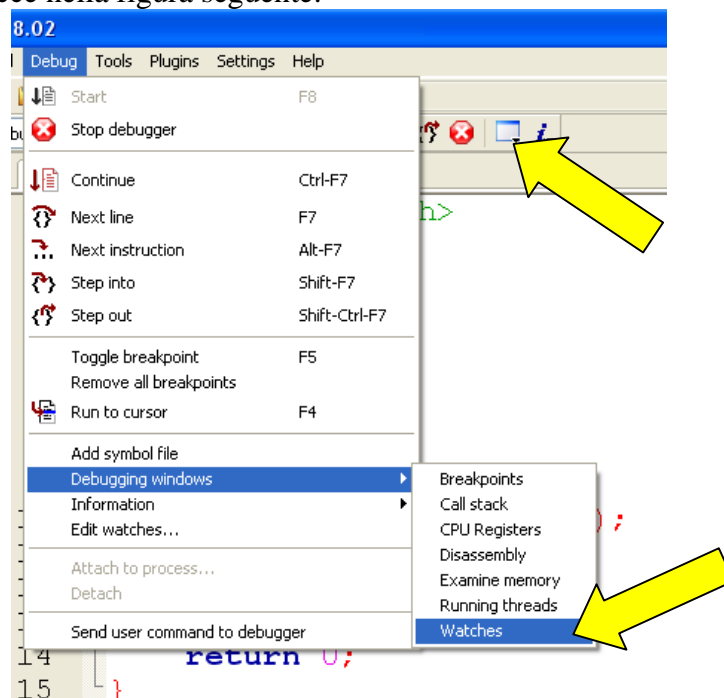
Quando una parte di codice è corretta e la si vuole eseguire senza interruzioni per iniziare l'esecuzione passo passo solo da un certo punto, si può impostare un breakpoint direttamente nella riga da cui iniziare l'esecuzione passo passo.

Alternativamente si posiziona il cursore sulla riga dalla quale iniziare l'esecuzione passo passo e si preme **F4** o si seleziona dal menu **Debug** la voce "**Run to cursor**".

Per impostare un breakpoint ad una determinata riga è sufficiente fare click nella zona grigia subito a destra dei numeri di riga, un punto rosso identifica la presenza di un breakpoint. Per eliminarlo si procede allo stesso modo. Ci possono essere più breakpoint nello stesso codice e per continuare velocemente l'esecuzione fino al successivo breakpoint si può usare **Control-F7 (Continue)**.

Per visualizzare il valore delle variabili se si è eseguito quanto indicato al punto 12 precedentemente è sufficiente posizionare (senza fare click) il mouse sul nome della variabile di cui vedere il valore.

La finestra dei *Watch* può essere utilizzata per visualizzare un elenco di variabili da controllare e il loro contenuto. Per attivarla si può fare click su uno dei due punti indicati dalle frecce nella figura seguente:



Si ottiene la finestra dei watch che conviene trascinare in basso a destra perché non sia flottante. Bisogna fare questa operazione una prima volta prima che il programma venga eseguito e una seconda volta mentre è in esecuzione passo passo. Il riquadro dove verrà posizionata viene visualizzato in azzurro:

