

## Índice

1Code::Blocks – Criação de projetos.....	1
2Code::Blocks – Localização do projeto.....	5
3Code::Blocks – Abertura de projetos já existentes.....	7
4Code::Blocks – Funcionamento.....	8
5Code::Blocks – Bibliotecas.....	10
6Code::Blocks – Múltiplos ficheiros.....	13
7Code::Blocks – Depuração de programas.....	19

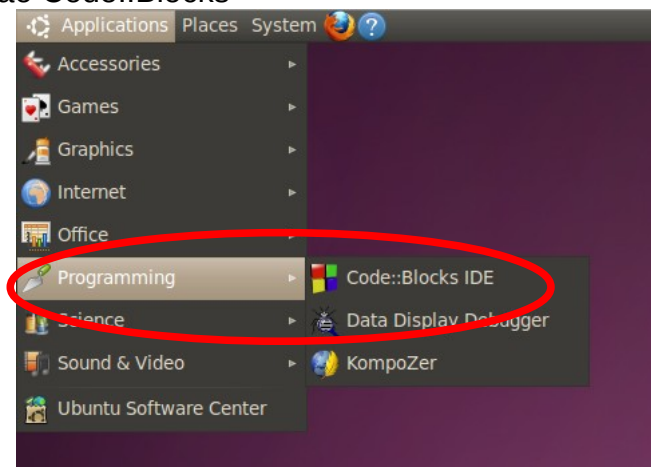
## 1 Code::Blocks – Criação de projetos

O Code::Blocks é um IDE (ambiente de desenvolvimento integrado) que permite a escrita do código, compilação, correção de erros e depuração integradas numa única aplicação.

Os vários ficheiro de código (.c e .h) que formarão uma aplicação são agrupados num projeto.

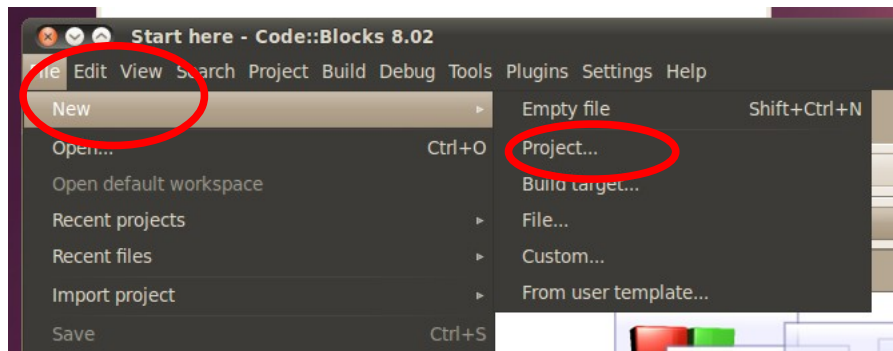
Para criar um projeto, é necessário seguir os seguintes passo:

- Abrir a aplicação Code::Blocks

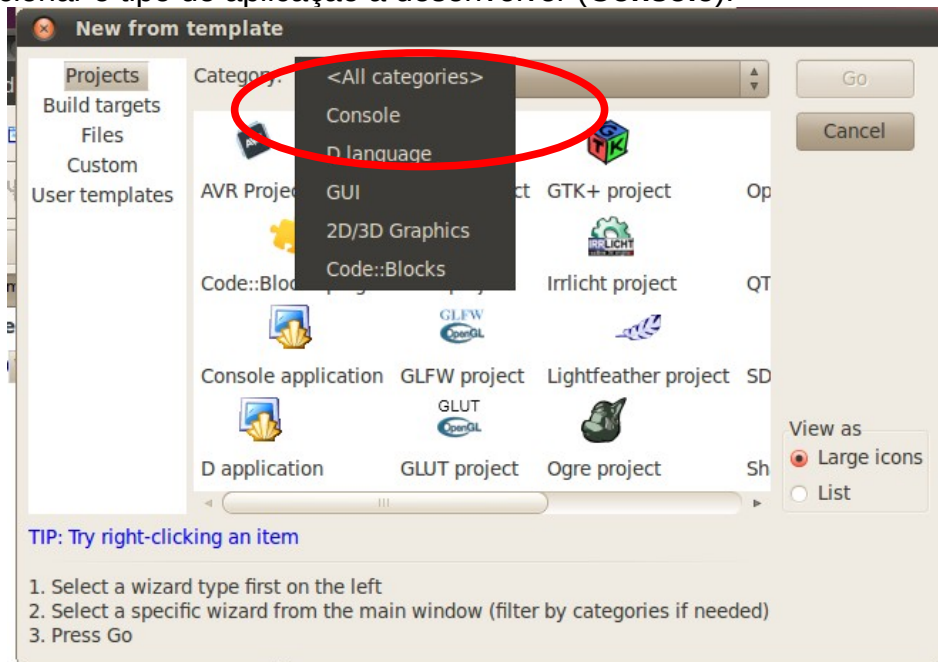


- Aceitar a mensagem inicial, carregando no botão **Close**

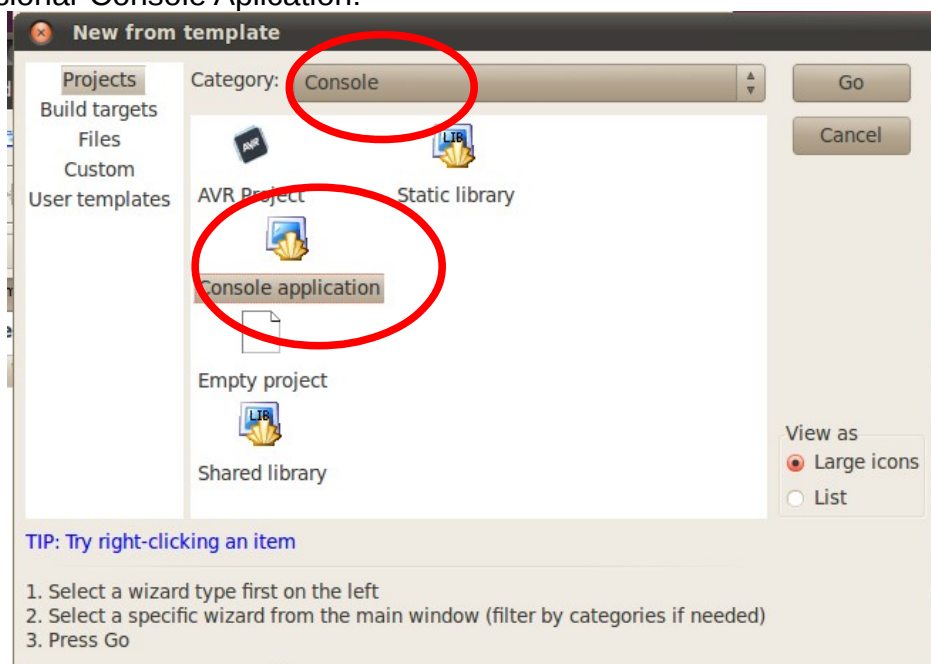




- Selecionar o menu **File** → **New** → **Project**
- Selecionar o tipo de aplicação a desenvolver (**Console**):



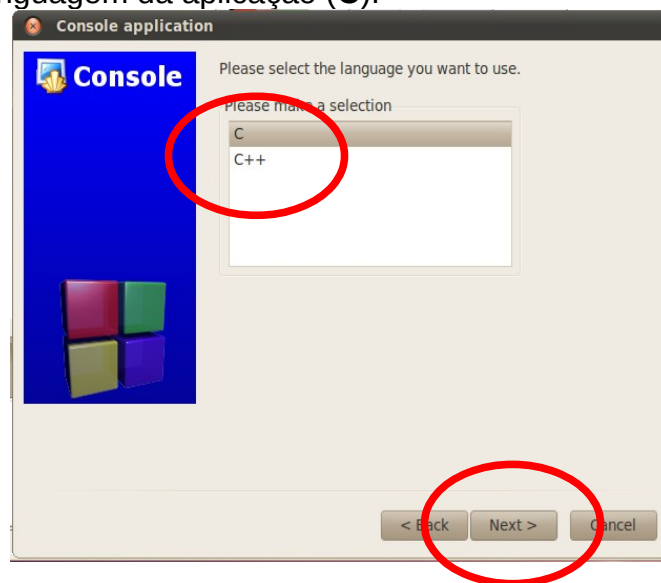
- Selecionar Console Application:



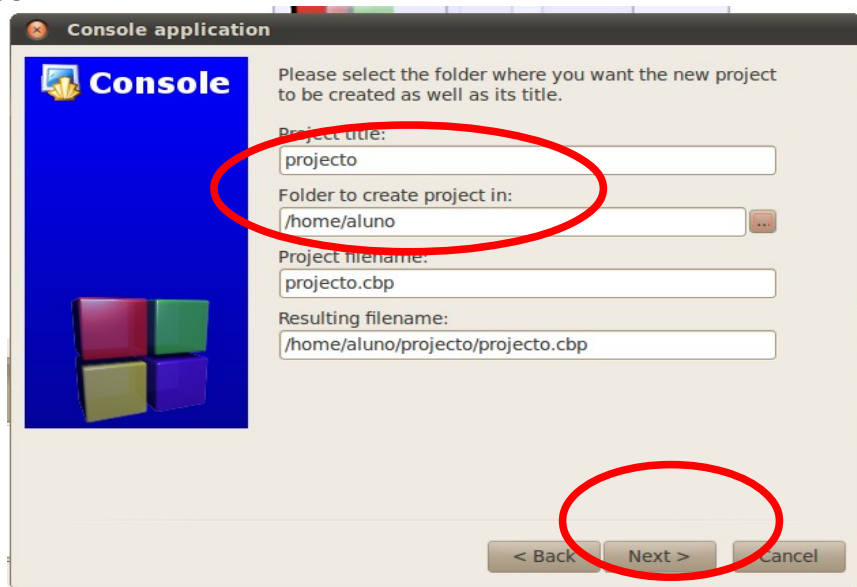
- Carregar no botão **Next**:



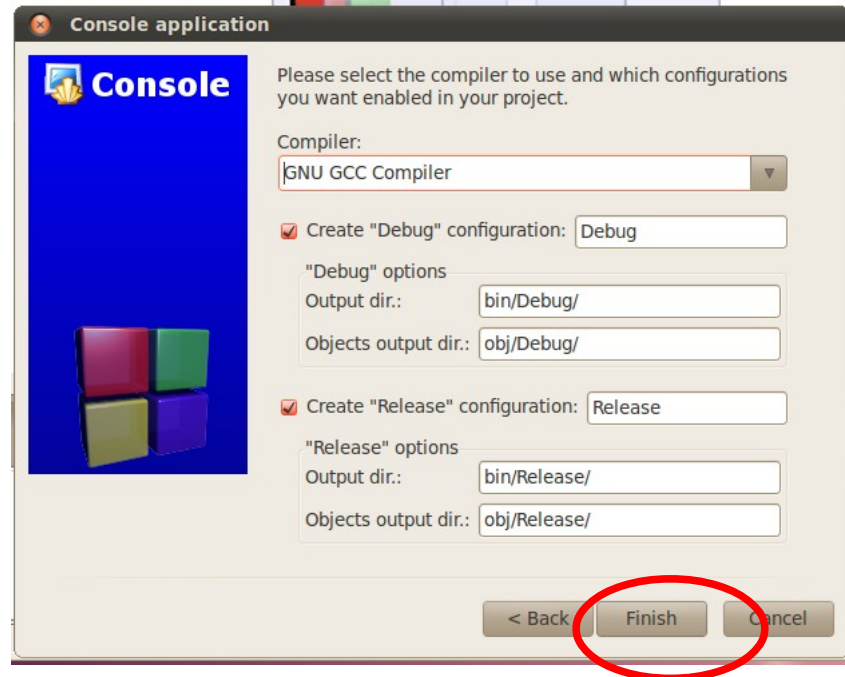
- Selecionar a linguagem da aplicação (C):



- Na janela seguinte deverá ser introduzido o nome do projeto. e a sua localização.
- Quer o nome do projeto. quer a diretoria não deverão ter espaços ou caracteres especiais.

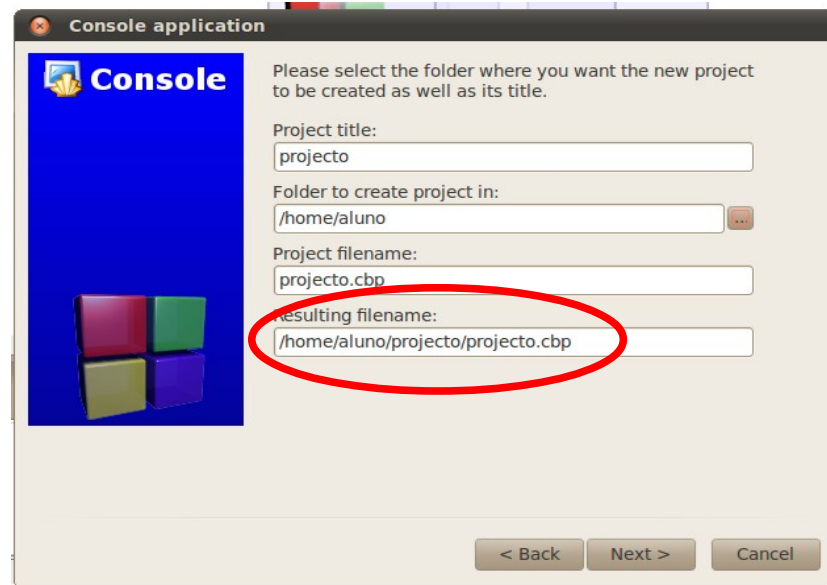


- Finalizar a criação do projeto. carregando no botão Finish:

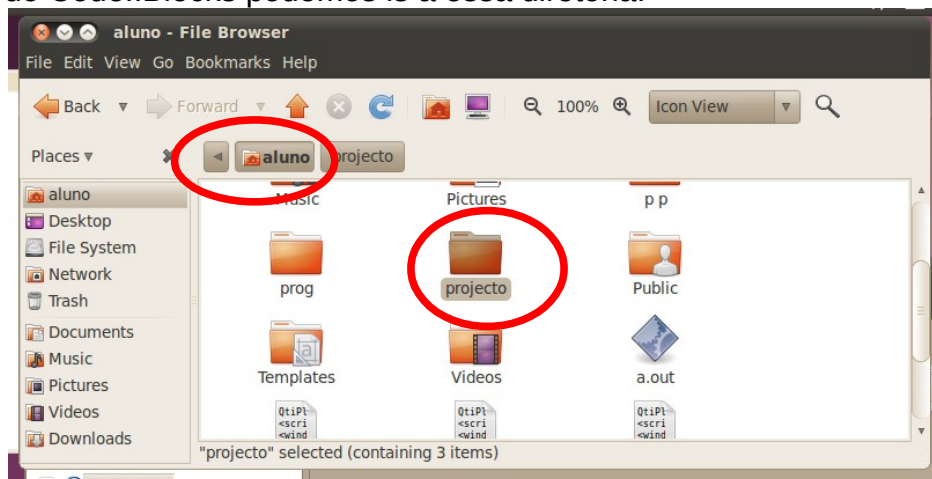


## 2 Code::Blocks – Localização do projeto.

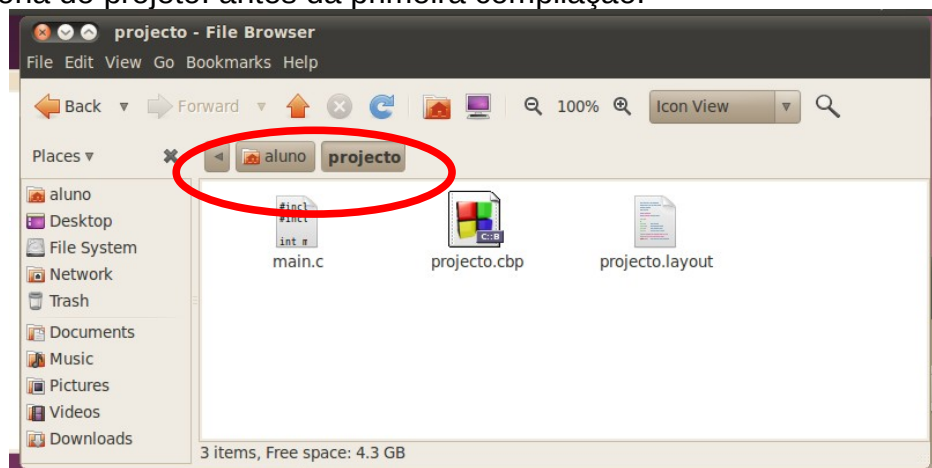
Cada projeto. criado no Code::Blocks é composto por uma série de ficheiros que se encontram localizados dentro de uma diretoria. No penúltimo passo da criação de um projeto. é indicada essa diretoria:



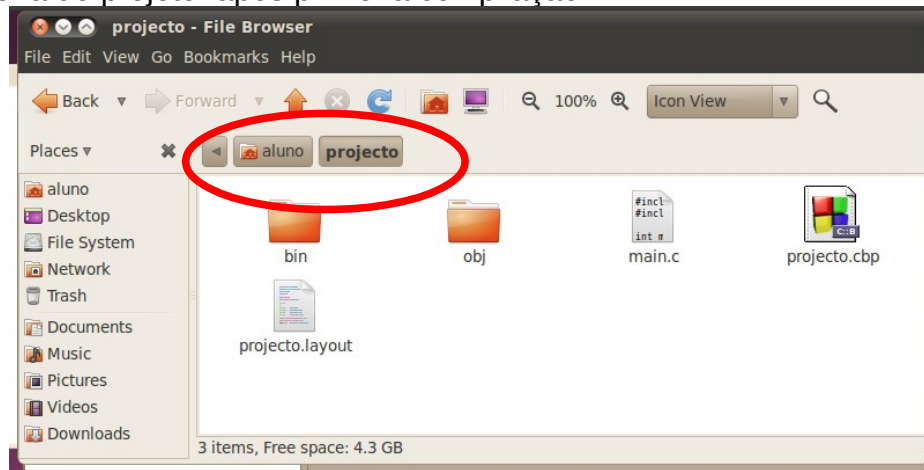
- Fora do Code::Blocks podemos ir a essa diretoria:



- Diretoria do projeto. antes da primeira compilação:



- Diretoria do projeto. após primeira compilação:

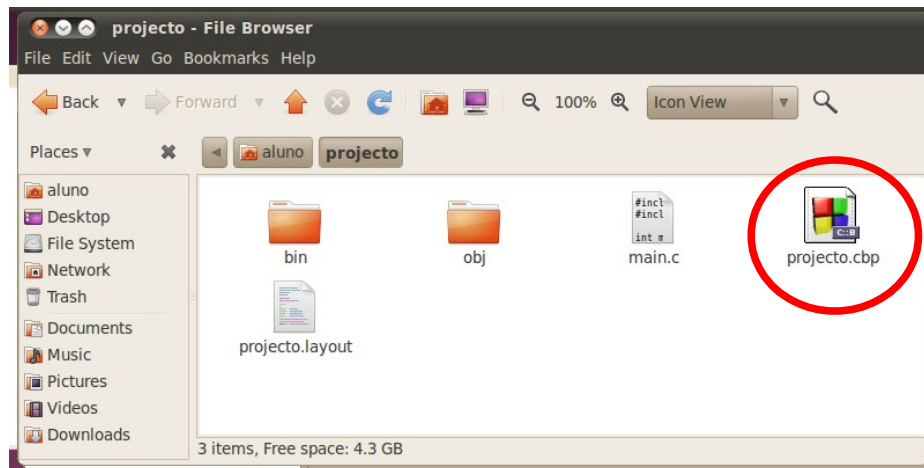


- Para transferir um projeto. para outro computador basta copiar a diretoria onde esse projeto. se encontra.

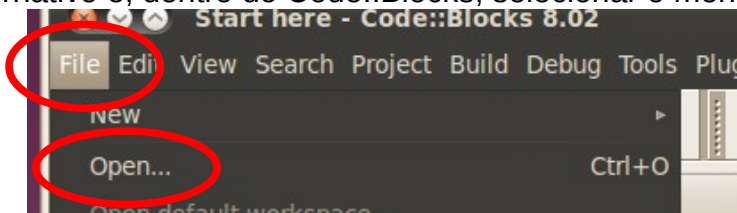
### 3 Code::Blocks – Abertura de projetos já existentes

Existem duas formas de abertura de um projeto já existente: através do File browser do Linux ou dentro do Code::Blocks

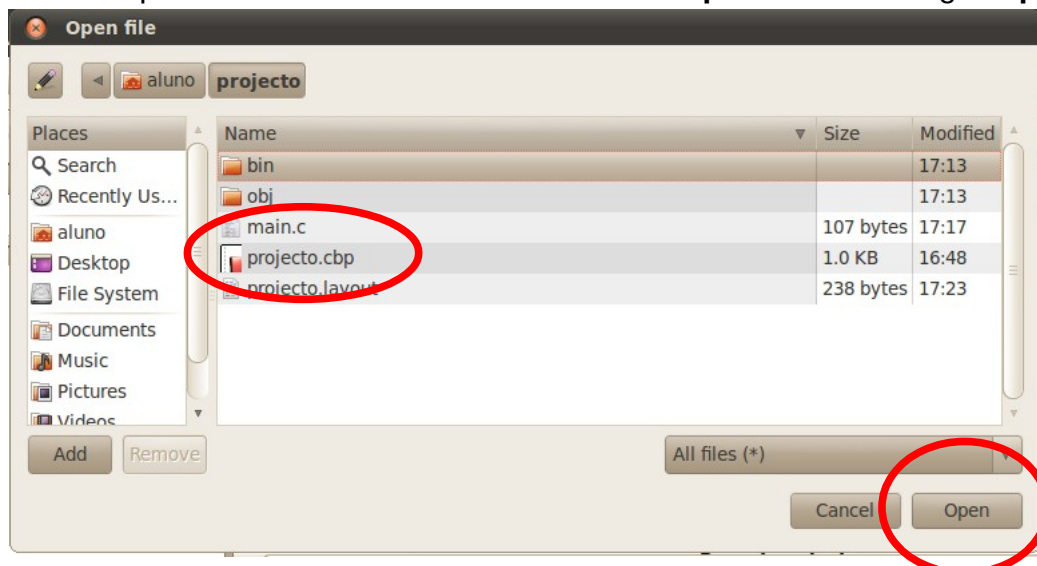
- Para abrir um projeto através do File browser basta aceder à pasta do projeto e fazer duplo clique sobre o ficheiro com a extensão **.cbp**



- A outra alternativa é, dentro do Code::Blocks, seleccionar o menus **File** → **Open**



- Procurar a pasta correta e seleccionar o ficheiro **.cbp** correto e carregar **Open**



- Após a abertura do projeto poderá ser necessário reconstruí-lo carregando no botão

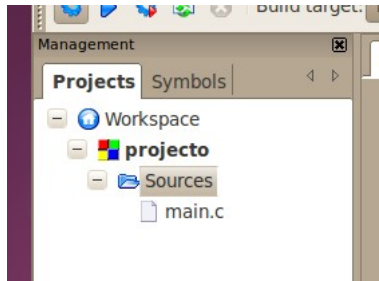




## 4 Code::Blocks – Funcionamento

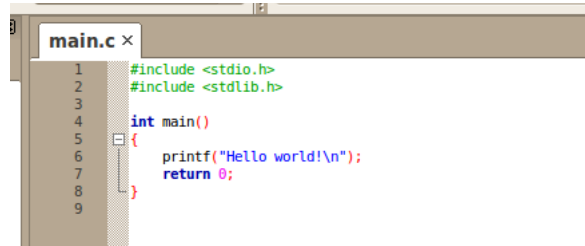
O Code::Blocks integra várias funcionalidade: editor, compilador com indicação dos erros , ambiente de execução das aplicações desenvolvidas e depurador (para encontrar os erros de execução).

No lado esquerdo da janela do Code::Blocks podem ser acedidos os vários ficheiros do projeto:



Na divisão **Sources** aparecerão os diversos ficheiros **.c** pertencentes ao projeto. Noutras divisões aparecem os ficheiros **.h**

Do lado direito da janela existe um editor normal:



Para compilar e executar a aplicação existem botões que invocam o compilador com os parâmetros adequados:



- compila todos os ficheiros que foram alterados. Antes de compilar grava todos os ficheiros pertencentes ao projeto que foram alterados.



- Executa a aplicação. Não compila a aplicação, sendo executada a última versão compilada



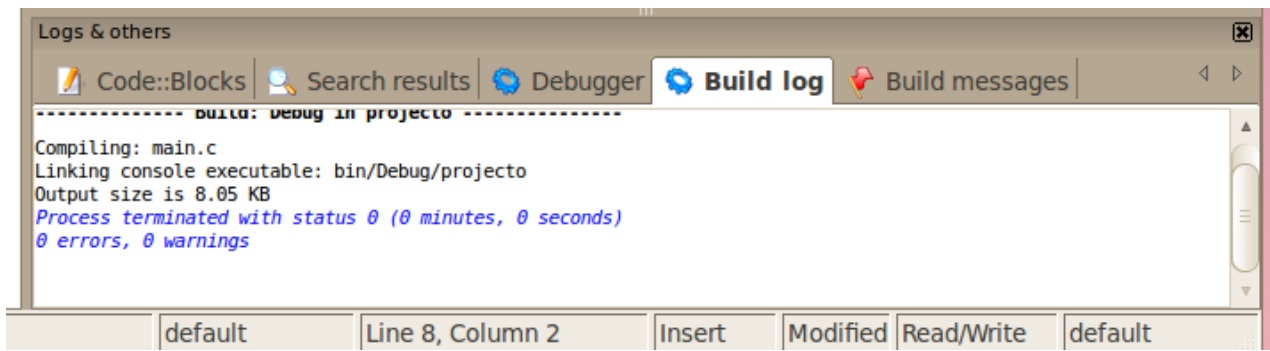
- Compila todos os ficheiros que foram alterados e executa o programa gerado



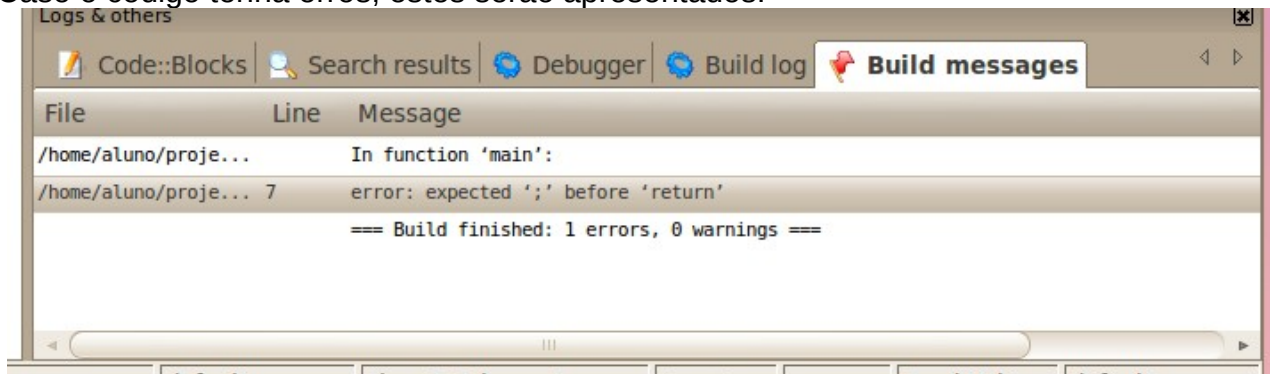
- Cria todos os ficheiros auxiliares, e compila a aplicação.

Se durante a compilação não forem detetados erros no fundo da janela aparecerá uma mensagem semelhante à seguinte:





Caso o código tenha erros, estes serão apresentados:




Clicando nessa linha, o cursor da janela de edição saltará para a linha com o erro.

## 5 Code::Blocks – Bibliotecas

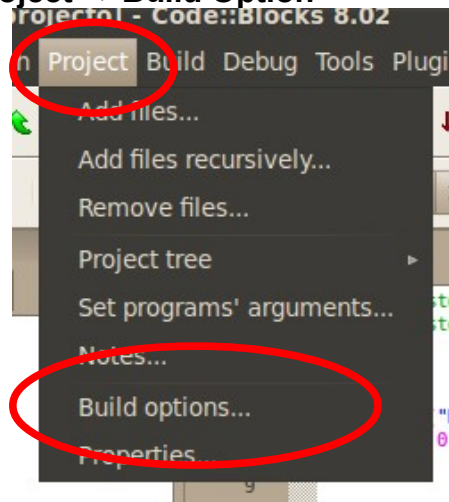
Diversas aplicações necessitam de bibliotecas externas: biblioteca matemática, biblioteca do G2, ...

Para usar essas bibliotecas é necessário dar ao gcc a indicação na necessidade dessas bibliotecas:

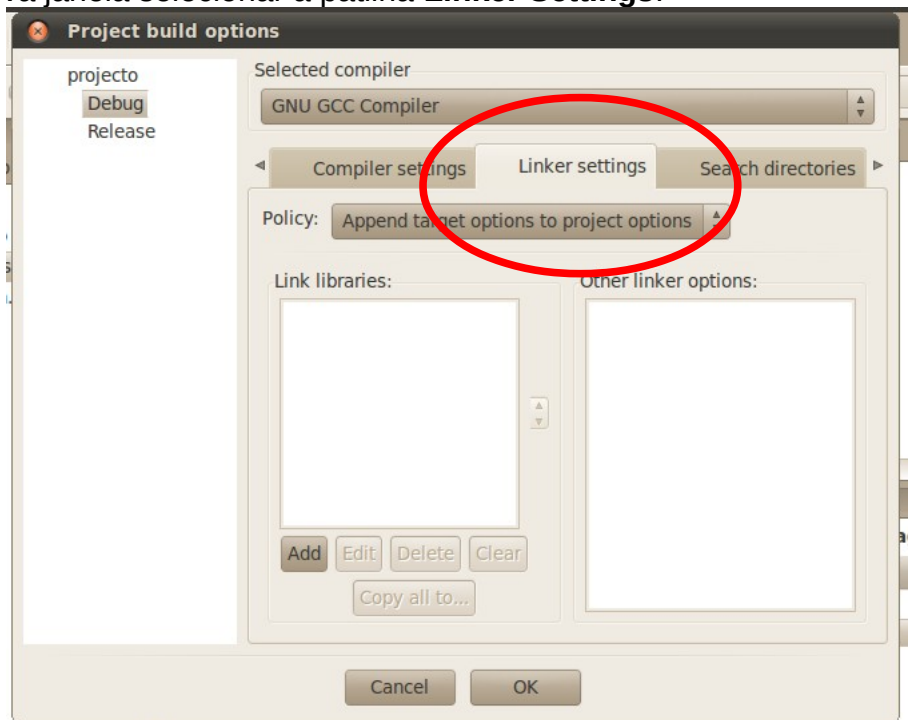
- biblioteca matemática **-lm**
- biblioteca do G2 **-lg2**
- biblioteca X11 **-lX11**
- biblioteca GD **-lgd**

O Code::Blocks facilita a vida do programador, ao permitir-lhe indicar uma única vez quais as bibliotecas que a aplicação necessita. A partir desse momento basta ao programador carregar no botão  sem se preocupar com os argumentos do gcc.

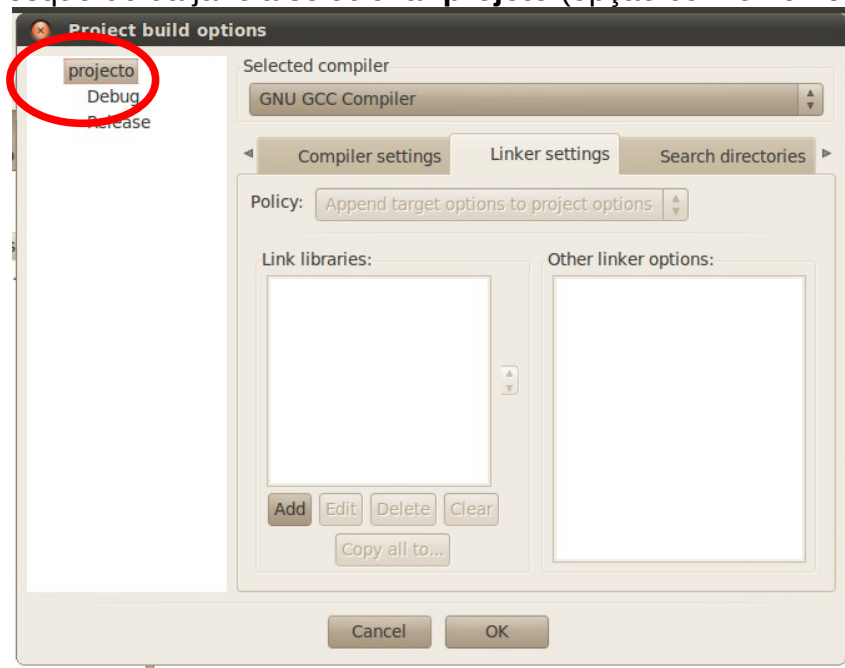
- Selecionar o menu **Project** → **Build Option**



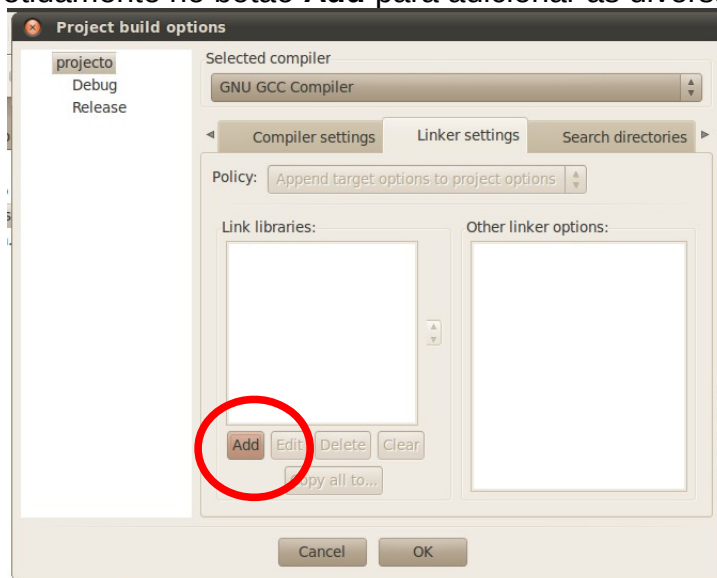
- Na nova janela selecionar a patilha **Linker Settings**:



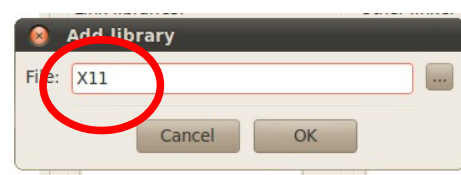
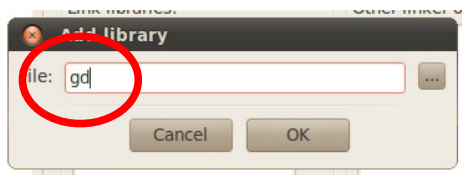
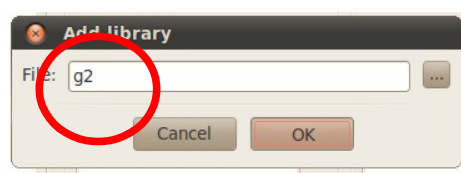
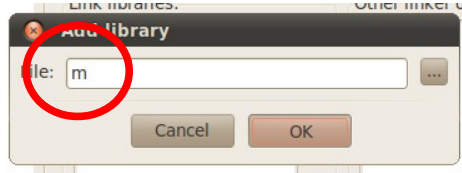
- No lado esquerdo da janela selecionar **projeto** (opção com o nome do projeto).



- Carregar repetidamente no botão **Add** para adicionar as diversas bibliotecas:

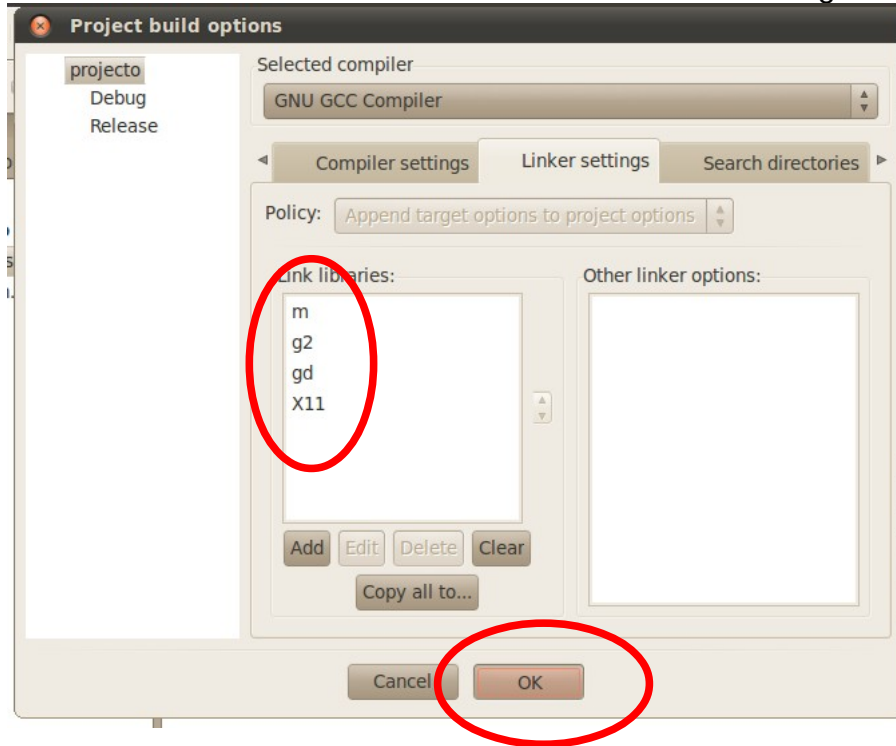


- Para cada biblioteca deverá ser adicionado o seu nome:



Atenção que apenas se deve introduzir o nome da biblioteca. Se na linha de comando usamos -lX11, aqui apenas deverá ser introduzido X11.

- Após todas as bibliotecas terem sido adicionadas deve-se carregar no botão **OK**



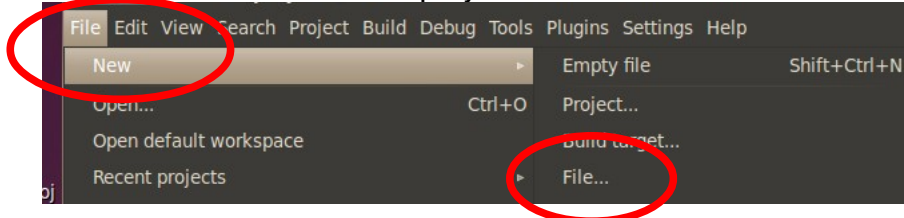
- Poderá ser necessário carregar no  botão de modo a reconstruir o projeto

## 6 Code::Blocks – Múltiplos ficheiros

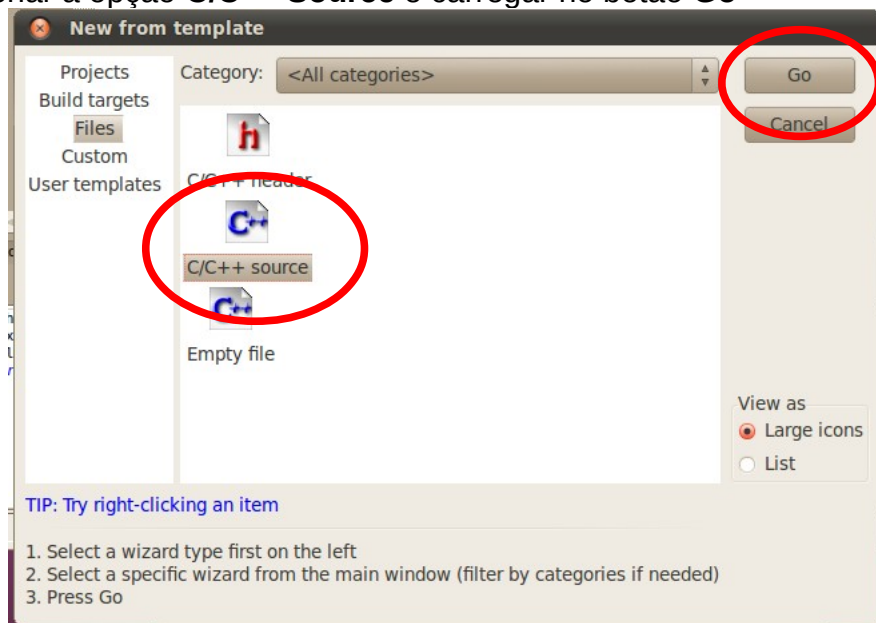
De modo a facilitar a codificação e estruturação de uma aplicação os projetos devem ter diversos ficheiros (.c e .h).

Em projetos com diversos ficheiros, o Code::Blocks gere automaticamente a sua compilação.

- Para introduzir um ficheiro .c num projeto seleccionar o menu **File** → **New** → **File**



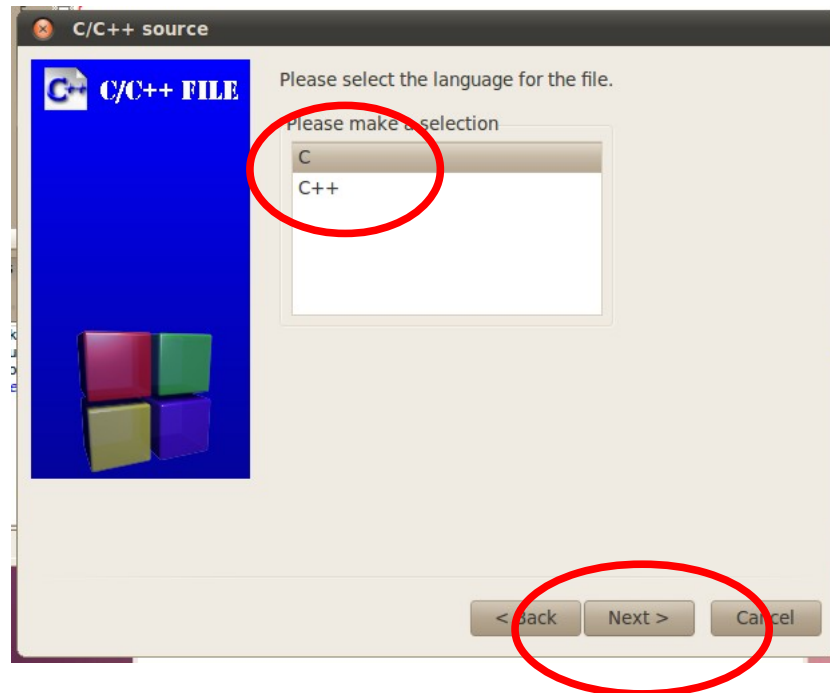
- Selecionar a opção **C/C++ Source** e carregar no botão **Go**



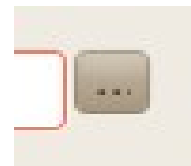
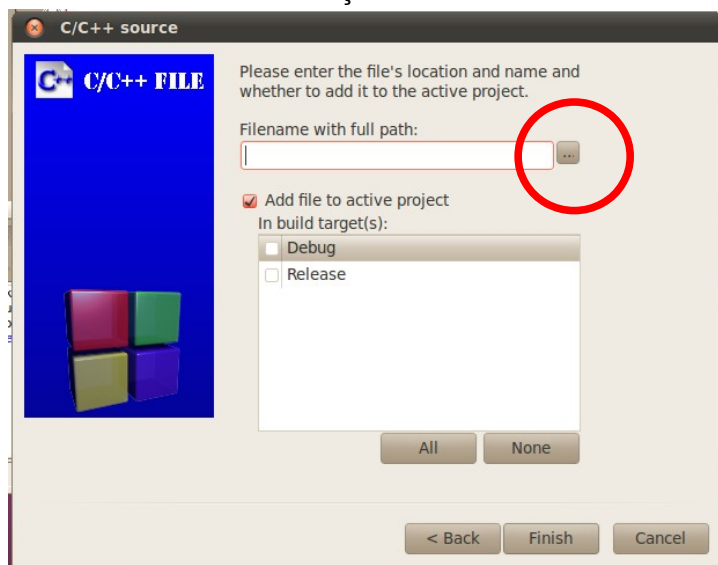
- Carregar em **Next**



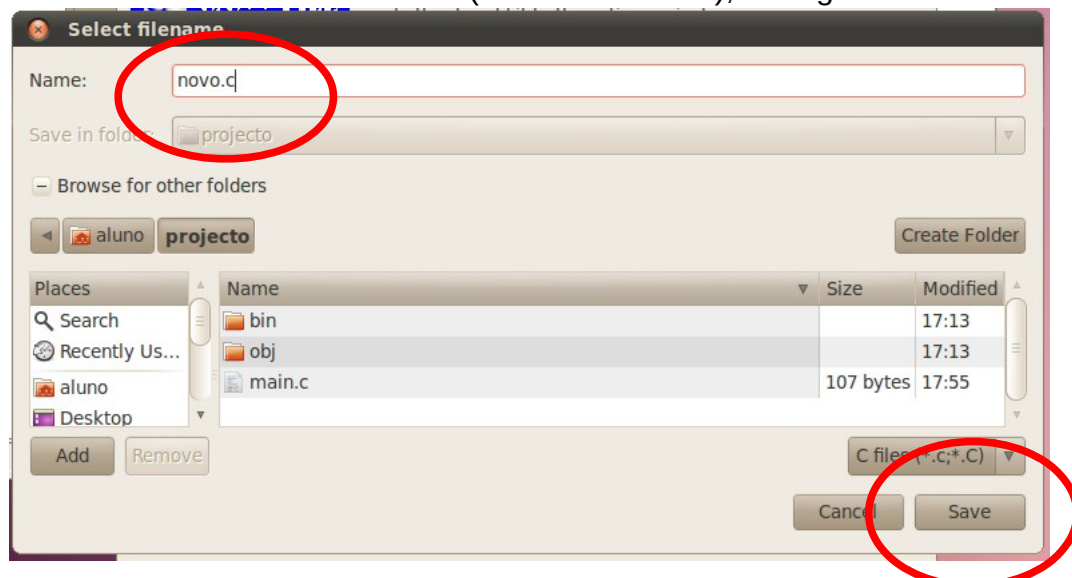
- Selecionar a linguagem **C** e carregar em **Next**



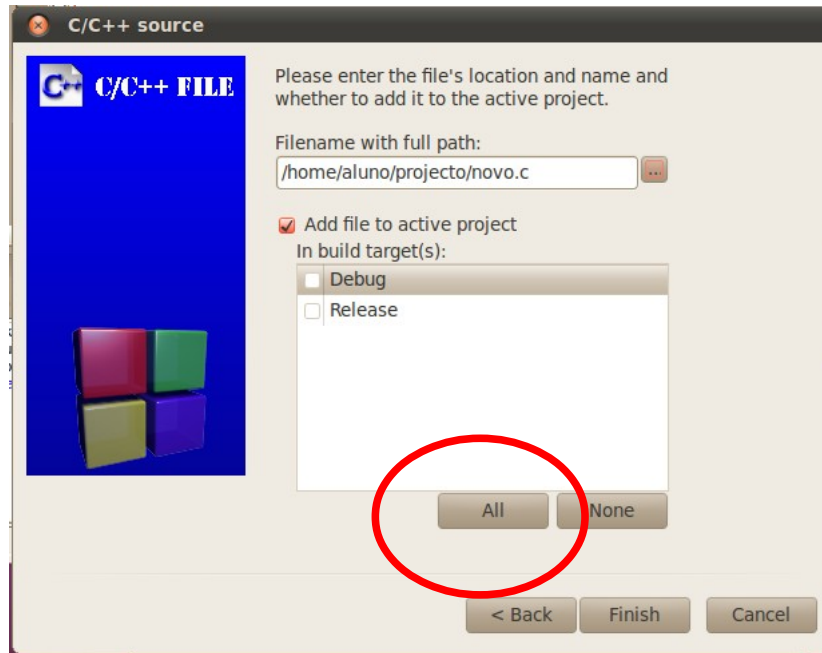
- Selecionar a localização e nome do ficheiro carregando no botão



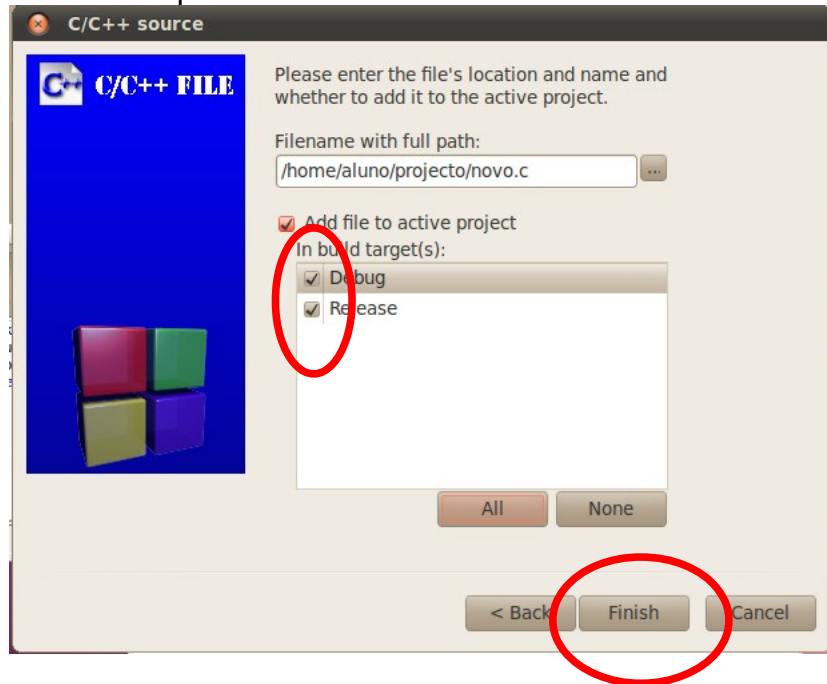
- Escrever o nome do novo ficheiro (com extensão **.c**), carregue em **Save**



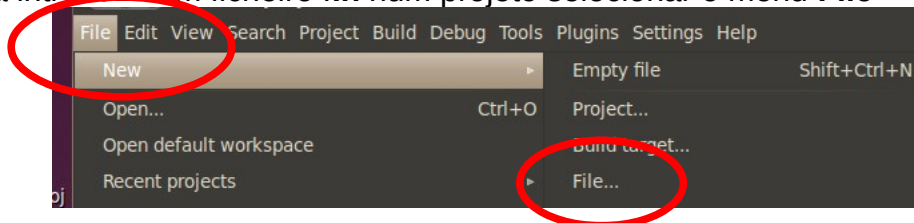
- Carregue em **All** para adicionar o ficheiro às versões de *Debug* e *Release*



- Carregue em **Finish** para terminar

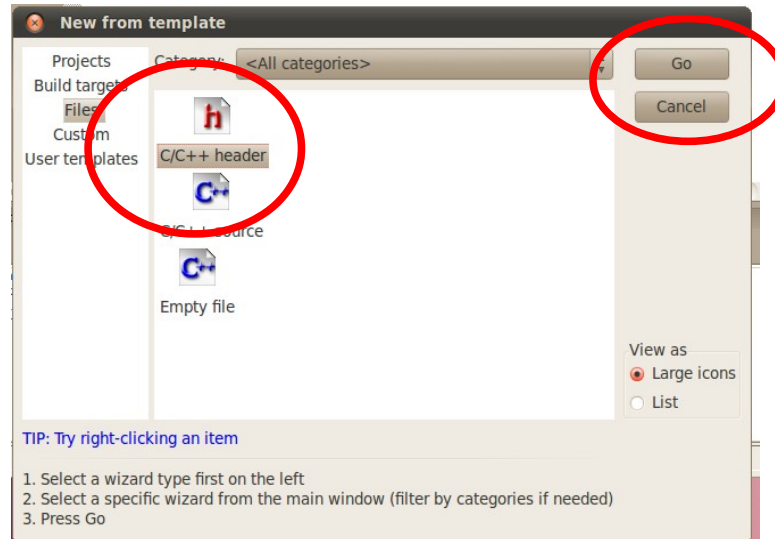


- Para introduzir um ficheiro **..h** num projeto seleccionar o menu **File** → **New** → **File**

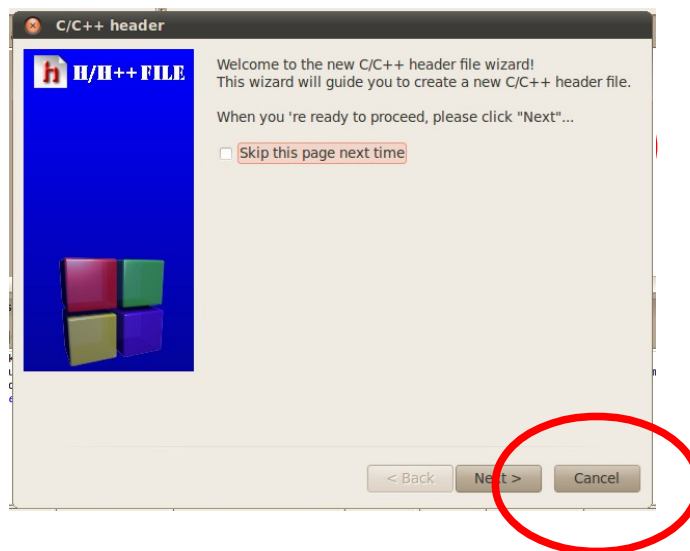




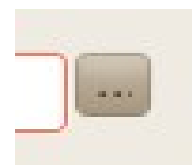
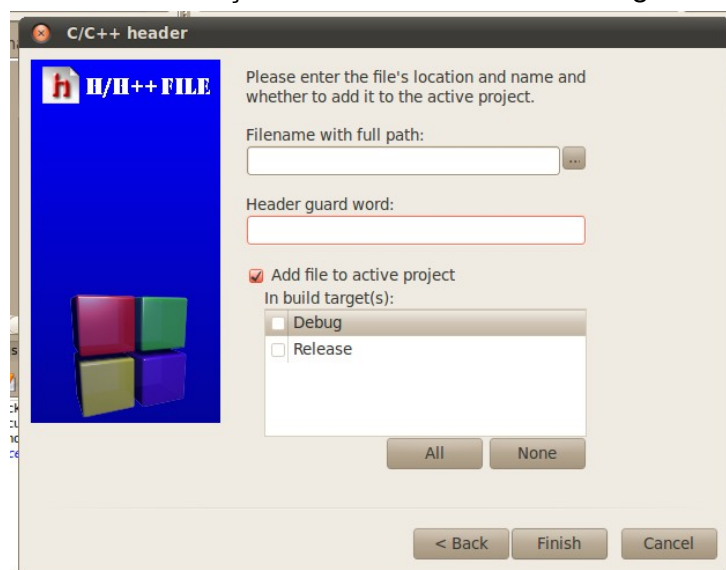
- Selecionar a opção **C/C++ Source** e carregar no botão **Go**



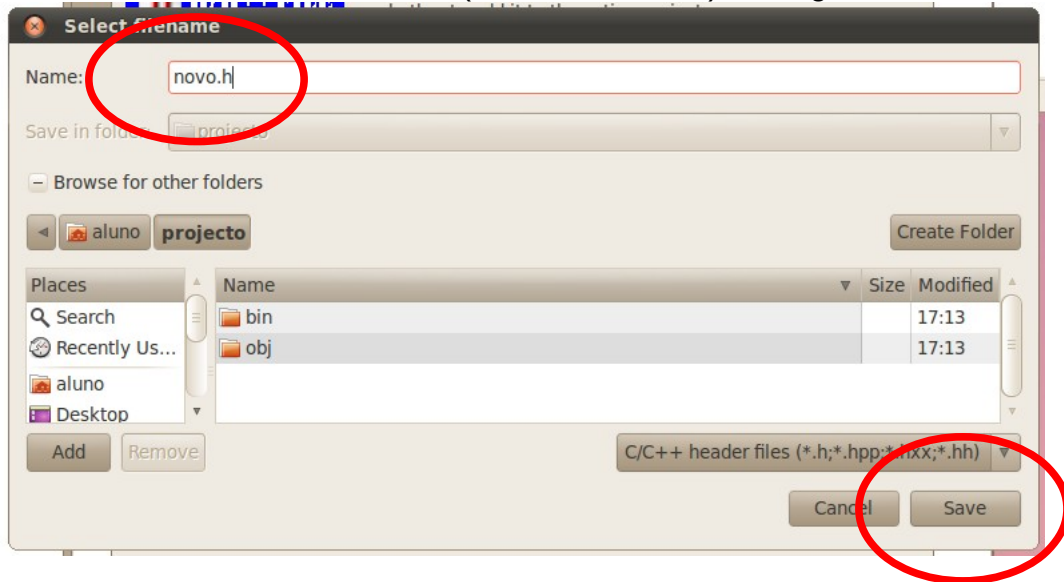
- Carregar em **Next**



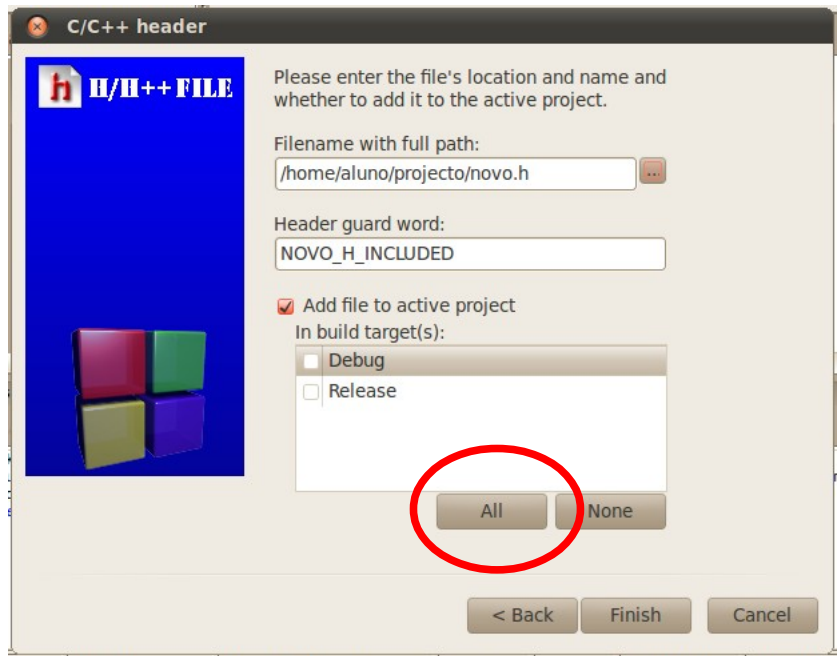
- Selecionar a localização e nome do ficheiro carregando no botão



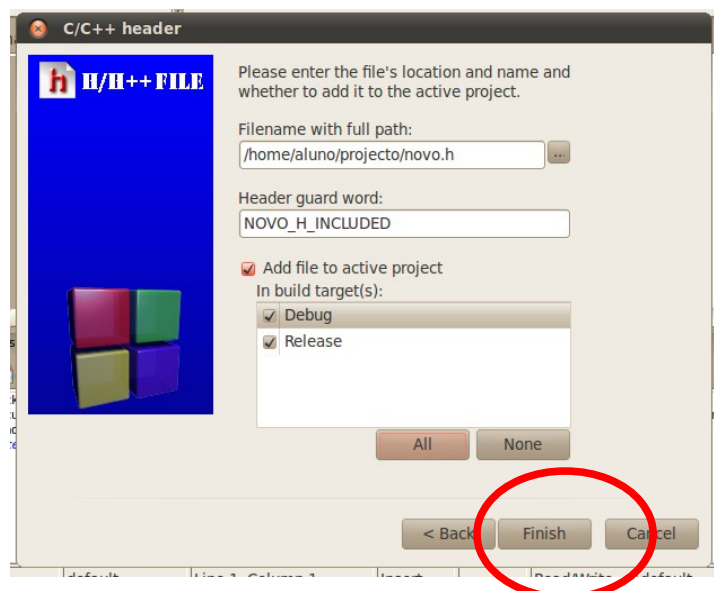
- Escrever o nome do novo ficheiro (com extensão **.h**), carregue em **Save**



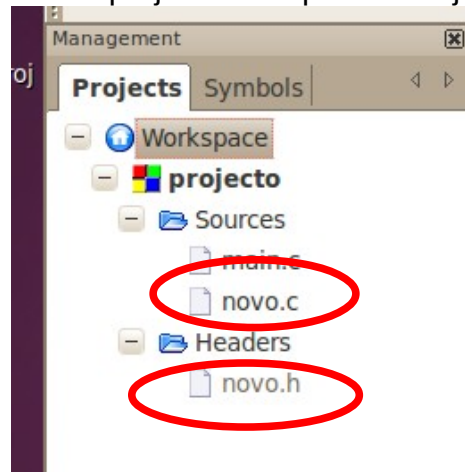
- Carregue em **All** para adicionar o ficheiro às versões de *Debug* e *Release*



Carregue em **Finish** para terminar



Após se adicionar um ficheiro a um projeto este aparece na janela principal.



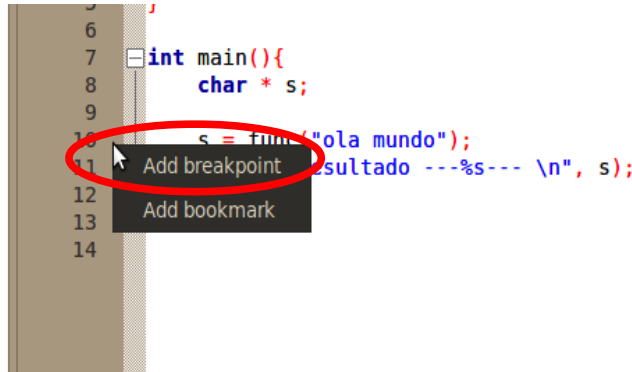
Fazendo duplo clique sobre esses novos ficheiros estes serão abertos no editor.

Se forem seguidos corretamente estes passo na próxima compilação também estes ficheiros serão considerados.

## 7 Code::Blocks – Depuração de programas

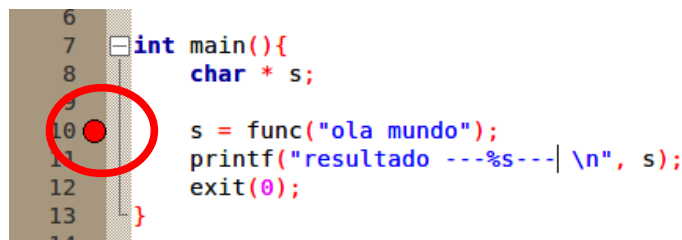
O Code::Blocks também integra as funcionalidades de um *debugger*, substituindo o DDD. Apesar dos princípios de funcionamento serem semelhantes aos do DDD, os ícones são diferente:

- Para marcar um *breakpoint* (local onde a execução da aplicação parará para observação do seu estado) é necessário carregar com o botão direito do rato na coluna contendo o número da linha:



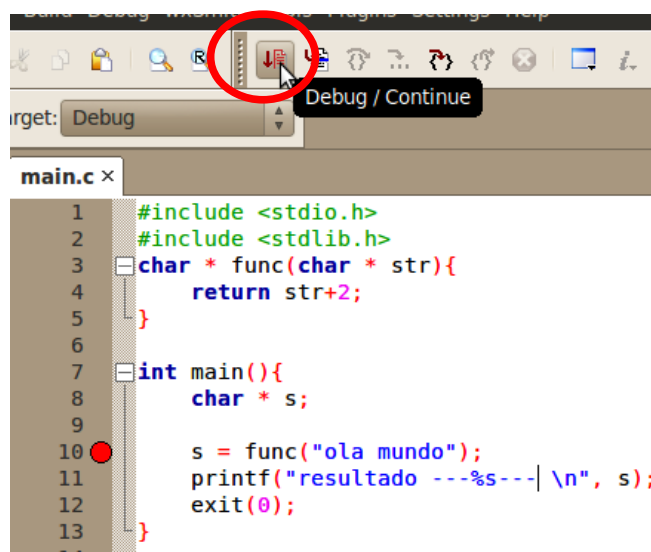
- Selecionar a opção **Add Breakpoint**.

Aparecerá então uma marca nessa linha:

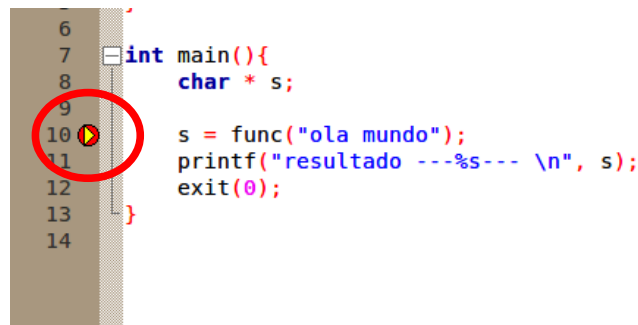


Pode-se então iniciar a aplicação dentro do debugger.

- Carregar no botão Debug/Continue:

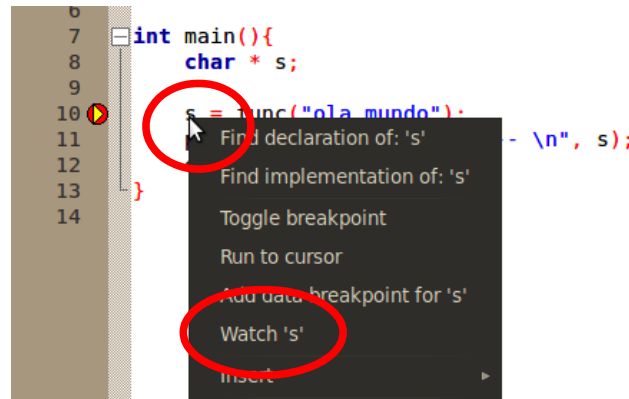


A aplicação executar-se-á até atingir a linha onde foi criado o *breakpoint*:



O valor das diversas variáveis pode ser visto na janela de **watch**. Para tal é necessário adicioná-las a essa janela:

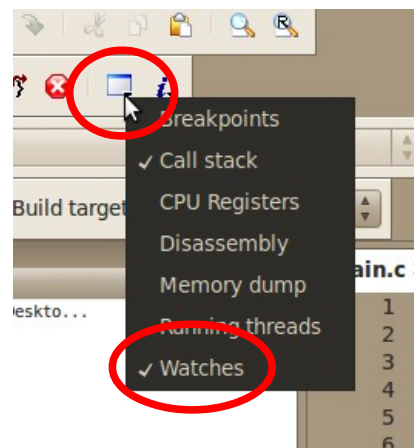
- Carregar com o botão direito do rato em cima da variável



- Escolhe a opção **Watch**

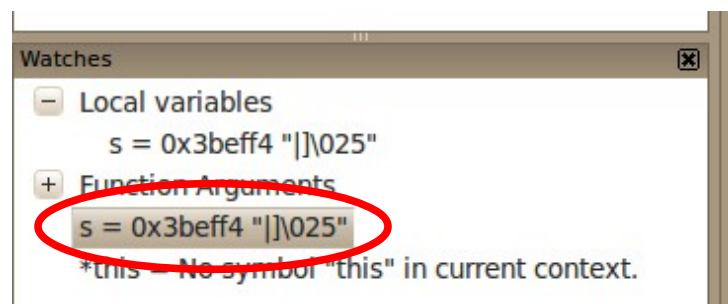
Se a janela **Watches** já estiver visível o valor desta variável aparecerá lá. Se a janela ainda não estiver visível é necessário abri-la:

- Carregar no botão **Debugging Windows**








- Ativar a opção **Watches**

O valor desta variável pode então ser consultado na janela **Watches**:



Para continuar a execução da aplicação podem-se usar os seguintes botões:

	Debug/Continue	Continua a execução da aplicação até terminar ou encontrar um breakpoint.
	Next Line	Executa a linha atual, parando na linha seguinte e "saltando por cima de funções".
	Step into	Executa a linha atual, parando numa linha de código diferente. Se a linha atual for uma função, entra e para na sua primeira linha de código.
	Step out	Continua a execução do código, parando apenas quando sair da função atual.
	Stop debugger	

```

5  }
6
7  int main(){
8      char * s;
9
10     s = func("ola mundo");
11     printf("resultado ---%s--- \n", s);
12     exit(0);
13 }
14

```