



UNIVERSIDADE  
FEDERAL DO CEARÁ



## **CK0179 – Programação Computacional para Engenharia: Utilizando a IDE**

Prof. Maurício Moreira Neto

# Objetivo

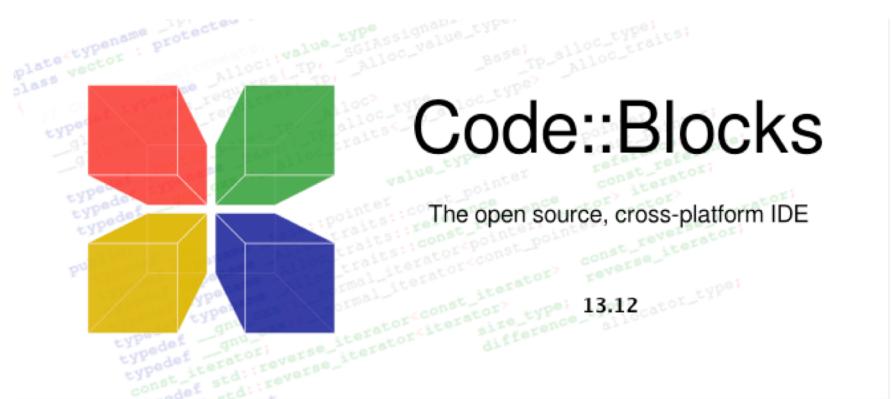
- Aprender sobre o que é uma IDE
- Aprender a utilizar uma IDE para criar e executar códigos na linguagem C
- Aprender quais são os recursos básicos de uma IDE para programação

# O que são IDEs?

- IDE significa Ambiente de Desenvolvimento Integrado (Do inglês, *Integrated Development Environment*)
- As IDEs podem ser utilizadas para programação em linguagem C e em diversas outras linguagens
- Um dos mais famosos utilizados na linguagem C é o Code::Blocks
  - IDE de código aberto e multiplataforma que suporta múltiplos compiladores

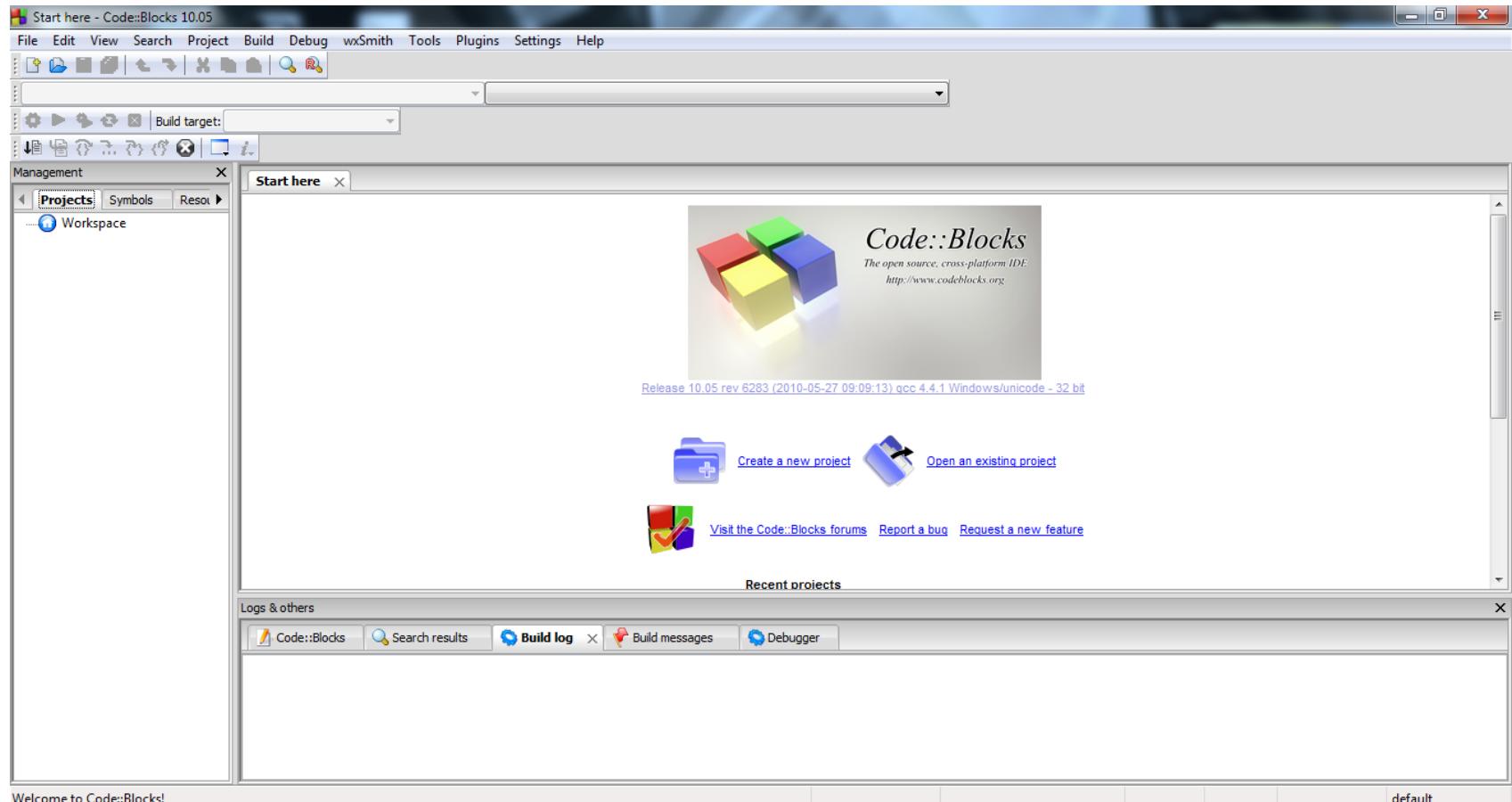
# O que é uma IDE?

- O Code::Blocks pode ser baixado diretamente pelo seu site: [www.codeblocks.org](http://www.codeblocks.org)
- Procure baixar a versão que inclui tanto a IDE do Code::Blocks como o compilador GCC e o debugger GDB da MinGW



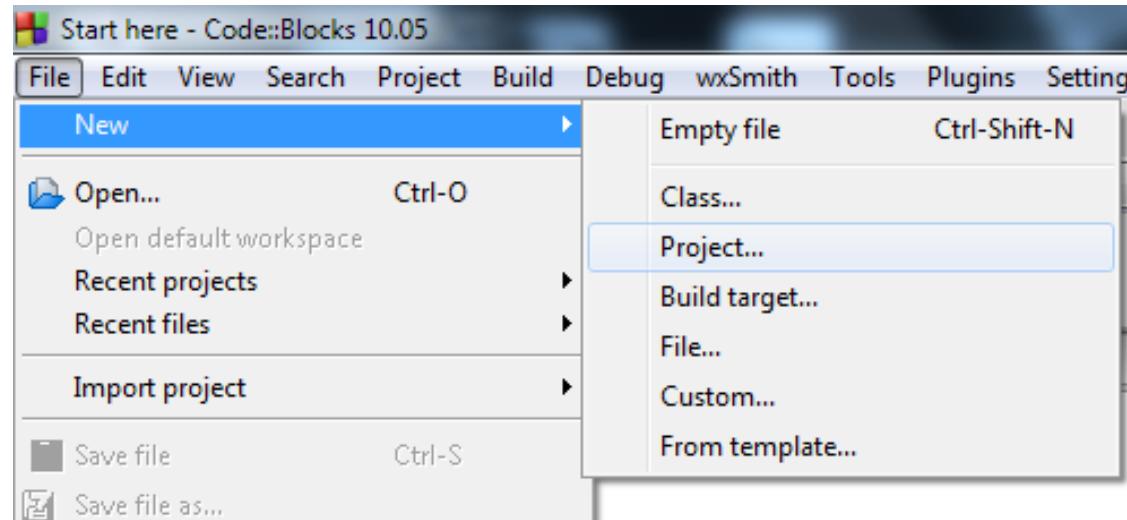
# Criando um projeto no Code::Blocks

- Primeiro, inicie o software Code::Blocks



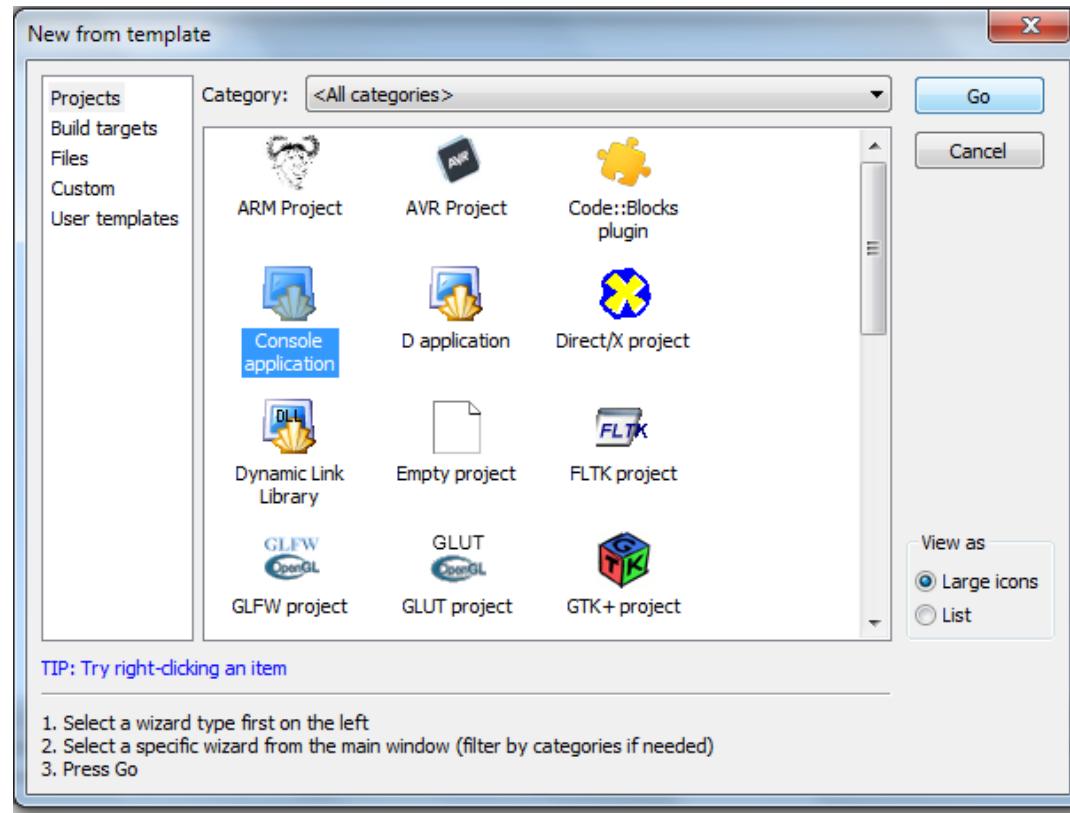
# Criando um projeto no Code::Blocks

- Em seguida clique em **File**, escolha **New** e depois **Project...**



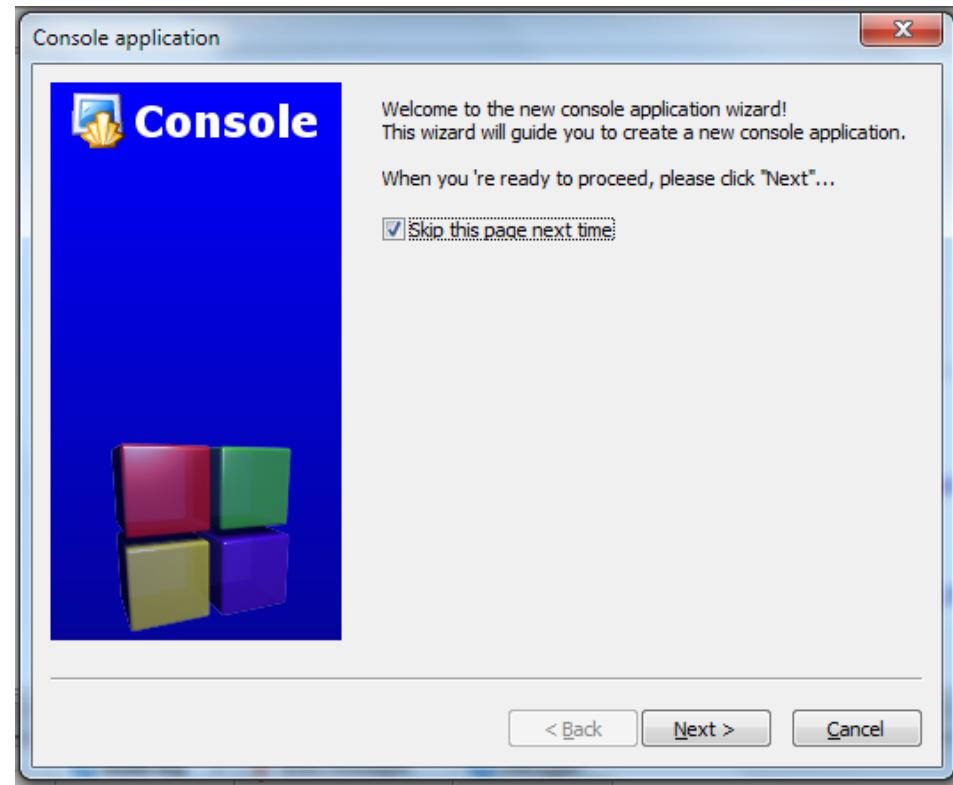
# Criando um projeto no Code::Blocks

- Uma lista de modelos (templates) de projetos vai aparecer. Escolha **Console Application**



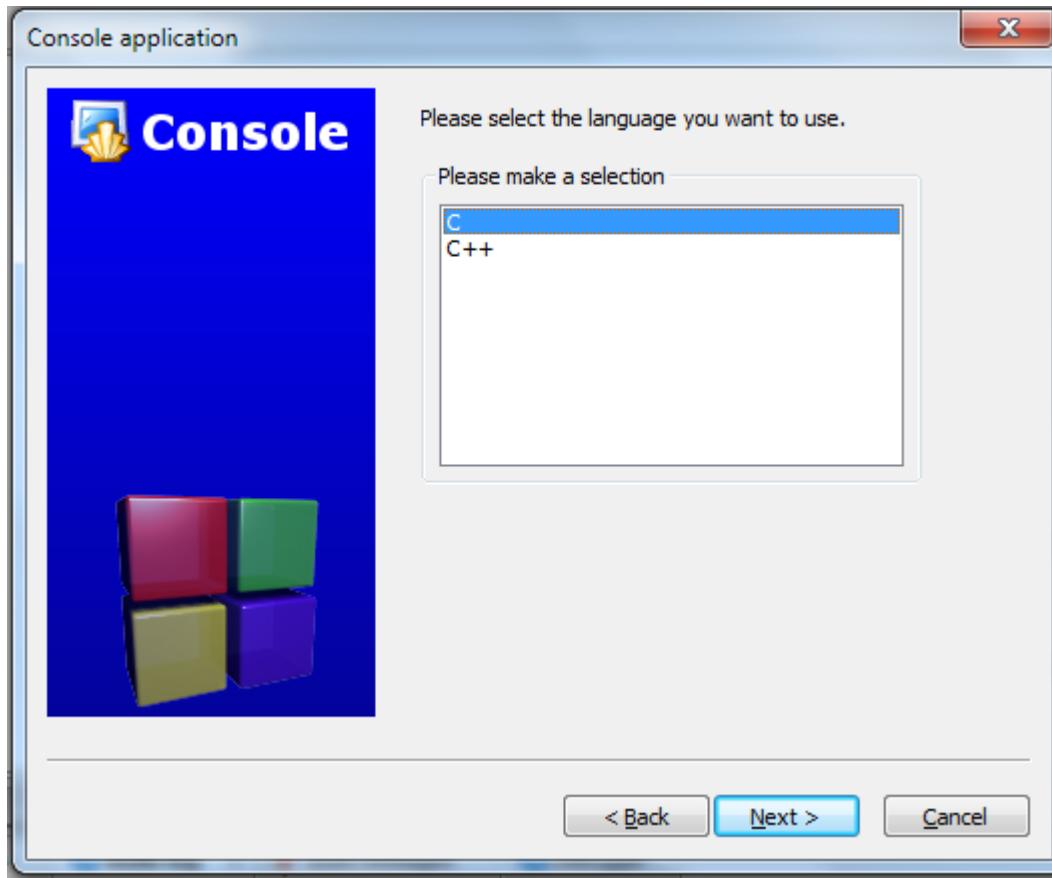
# Criando um projeto no Code::Blocks

- Caso esteja criando um projeto pela primeira vez, a tela a seguir vai aparecer
  - Se marcarmos a opção **Skip this page next time**, essa tela de boas-vindas não será mais exibida da próxima vez que criarmos um projeto
- Em seguida, clique em **Next**



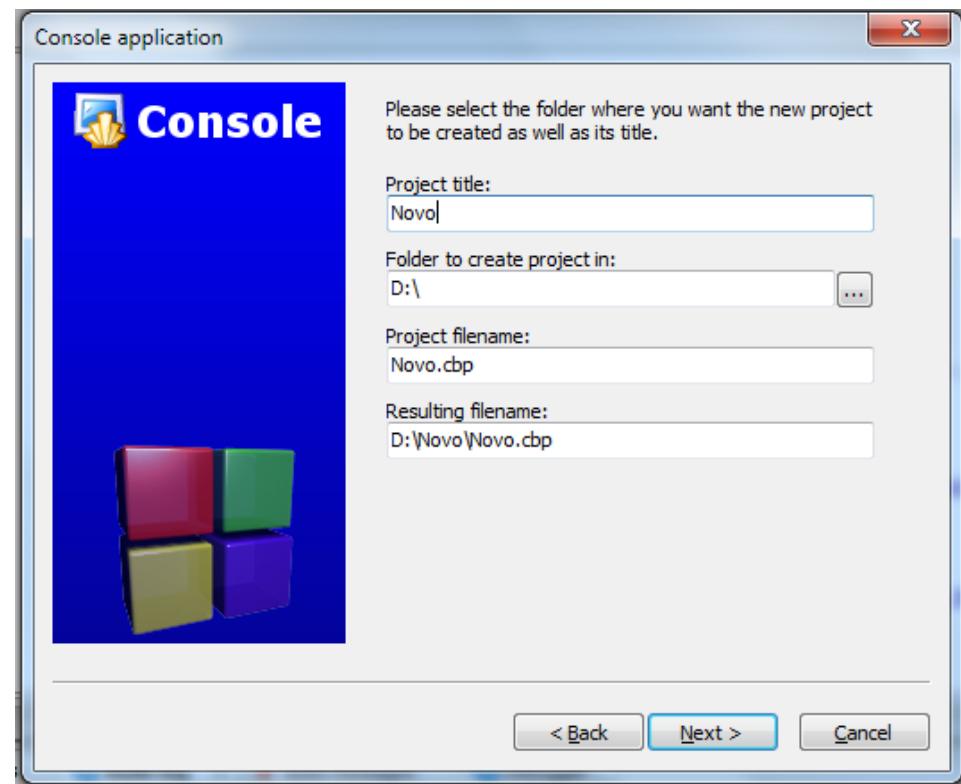
# Criando um projeto no Code::Blocks

- Escolha a opção C e clique em **Next**



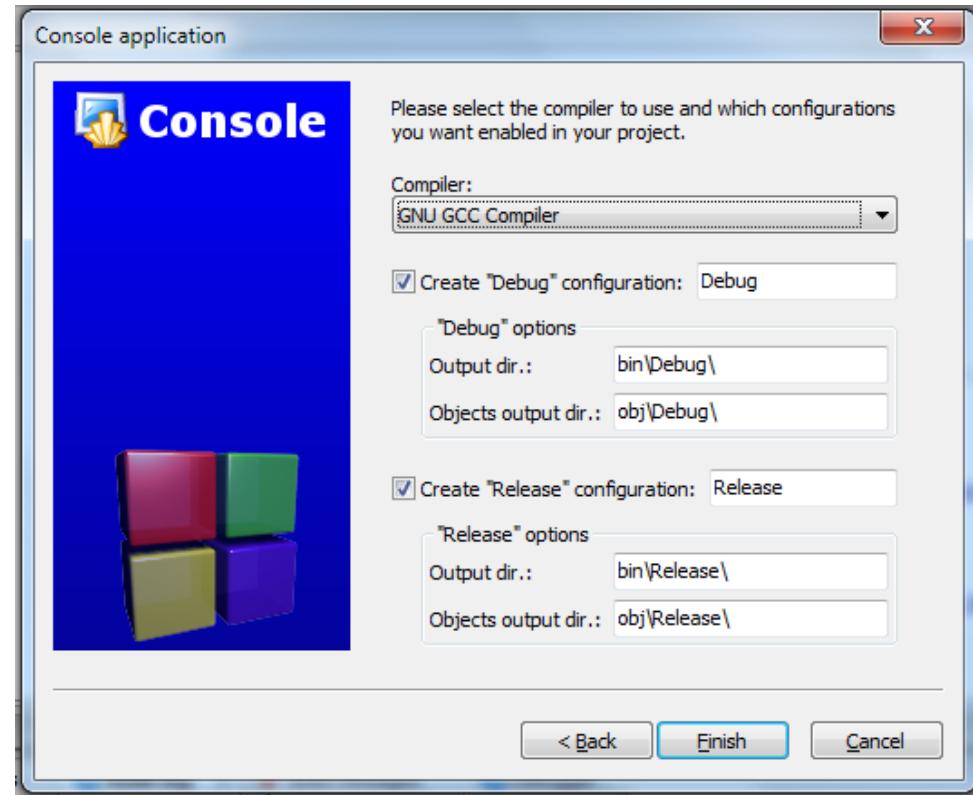
# Criando um projeto no Code::Blocks

- No campo **Project Title**, coloca-se um nome para o seu projeto. No campo **Folder to create Project in** é possível selecionar onde o projeto será salvo no computador
  - Evite espaços e acentuações no nome e caminho do projeto
- Clique em **Next** para continuar



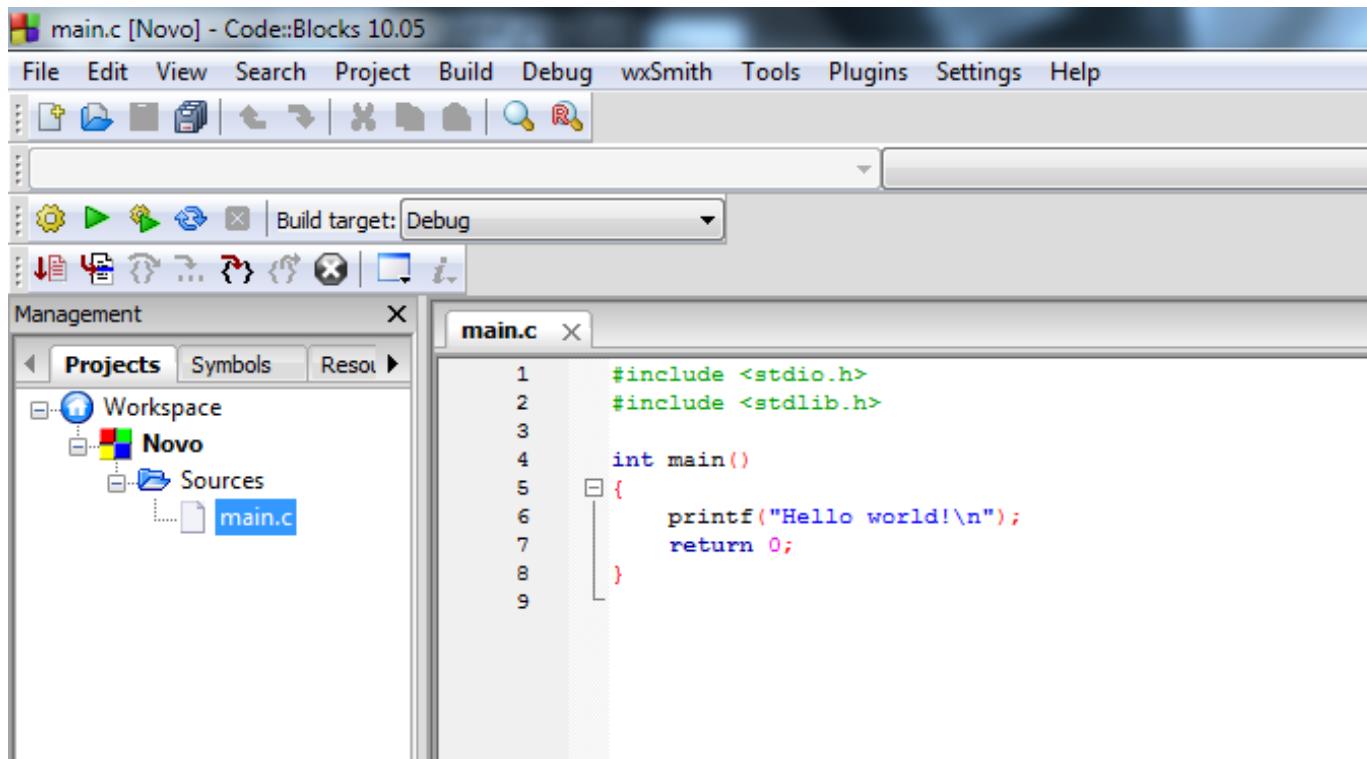
# Criando um projeto no Code::Blocks

- Na tela a seguir, algumas configurações do compilador podem ser modificadas
  - No entanto, isso não será necessário
  - Basta clicar em **Finish**



# Criando um projeto no Code::Blocks

- Ao fim destes passos, o esqueleto de um novo programa em linguagem C terá sido criado



# Criando um projeto no Code::Blocks

- Por fim, pode-se utilizar as seguintes opções do menu **Build** para compilar e executar nosso programa
  - **Compile current file (Ctrl + Shift + F9)**
    - Opção que transforma seu arquivo de código-fonte em instruções de máquina e gera um arquivo do tipo objeto
  - **Build (Ctrl + F9)**
    - Será compilado todos os arquivos do seu projeto para fazer o processo de “linkagem” com tudo o que é necessário para gerar o executável do seu programa
  - **Build and run (F9)**
    - Além de gerar o executável, essa opção também executa o programa gerado

# Utilizando o Debugger do Code::Blocks

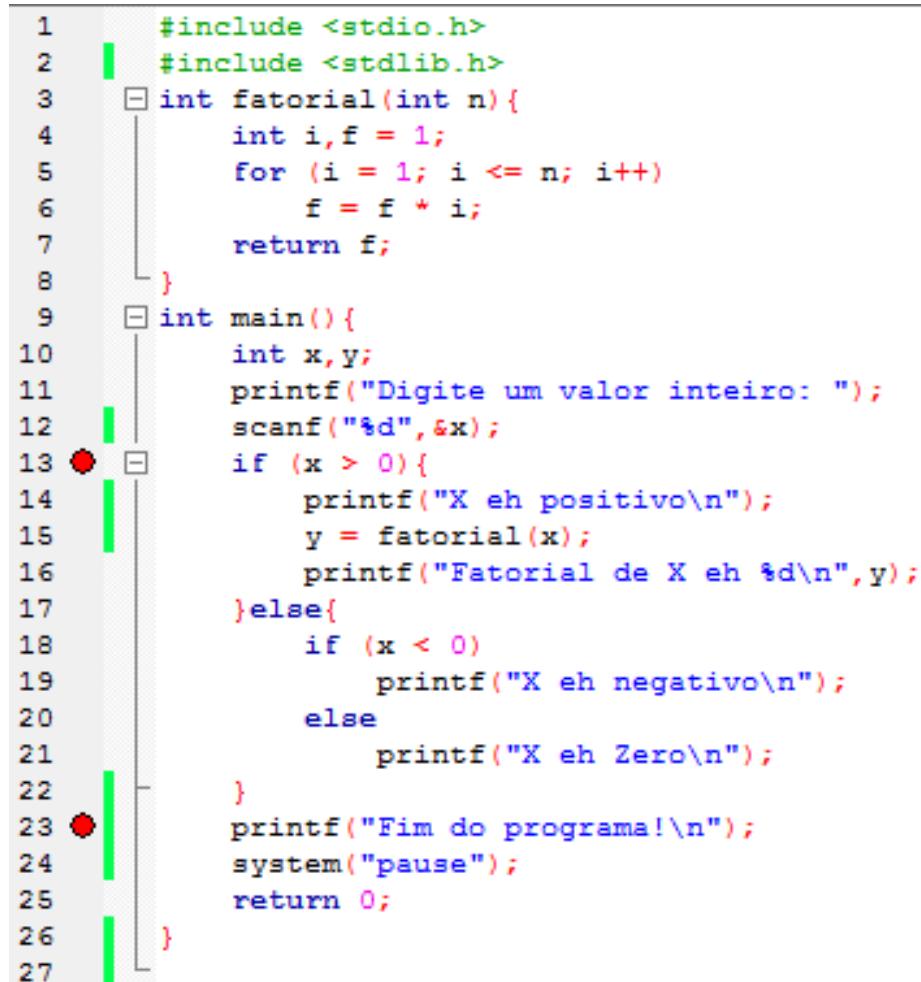
- Com o passar do tempo, nosso conhecimento sobre programação cresce, assim como a complexidade de nossos programas
- Surge a necessidade de examinar o nosso programa à procura de erros ou defeitos no código-fonte
- Para realizar essa tarefa, contamos com a ajuda de um **debugger** ou **depurador**

# Utilizando o Debugger do Code::Blocks

- O debugger nada mais é que um programa de computador usado para testar e depurar (limpar) outros programas
- Entre as principais funcionalidades de um debugger estão:
  - Possibilidade de executar o programa passo a passo
  - Pausar o programa em pontos predefinidos, chamados pontos de paradas (**breakpoints**), para examinar o estado atual de suas variáveis
- Todas as funcionalidades do debugger podem ser encontradas no menu **Debug**

# Utilizando o Debugger do Code::Blocks

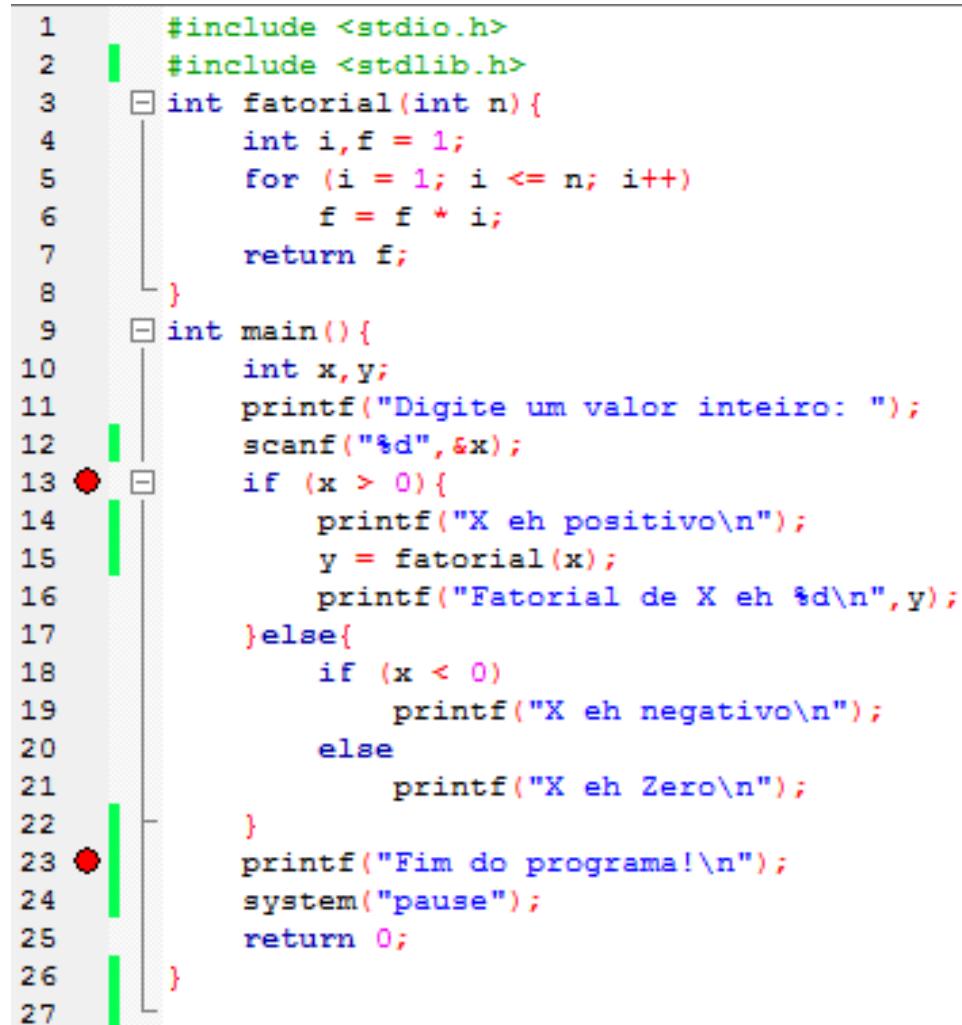
- Para utilizar o debugger do Code::Blocks, imagine o código ao lado
- Primeiramente, vamos colocar dois pontos de parada no programa, nas linhas 13 e 23
  - Isso pode ser feito clicando no lado direito do número da linha



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 int factorial(int n){
4     int i, f = 1;
5     for (i = 1; i <= n; i++)
6         f = f * i;
7     return f;
8 }
9 int main(){
10    int x, y;
11    printf("Digite um valor inteiro: ");
12    scanf("%d", &x);
13    if (x > 0){
14        printf("X eh positivo\n");
15        y = factorial(x);
16        printf("Fatorial de X eh %d\n", y);
17    }else{
18        if (x < 0)
19            printf("X eh negativo\n");
20        else
21            printf("X eh Zero\n");
22    }
23    printf("Fim do programa!\n");
24    system("pause");
25    return 0;
26 }
27 }
```

# Utilizando o Debugger do Code::Blocks

- Iniciamos o debugger com a opção **Start (F8)**
  - Isso fará com que o programa seja executado normalmente até encontrar um breakpoint



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 int factorial(int n){
4     int i, f = 1;
5     for (i = 1; i <= n; i++)
6         f = f * i;
7     return f;
8 }
9 int main(){
10     int x, y;
11     printf("Digite um valor inteiro: ");
12     scanf("%d", &x);
13     if (x > 0){
14         printf("X eh positivo\n");
15         y = factorial(x);
16         printf("Fatorial de X eh %d\n", y);
17     }else{
18         if (x < 0)
19             printf("X eh negativo\n");
20         else
21             printf("X eh Zero\n");
22     }
23     printf("Fim do programa!\n");
24     system("pause");
25     return 0;
26 }
27 }
```

# Utilizando o Debugger do Code::Blocks

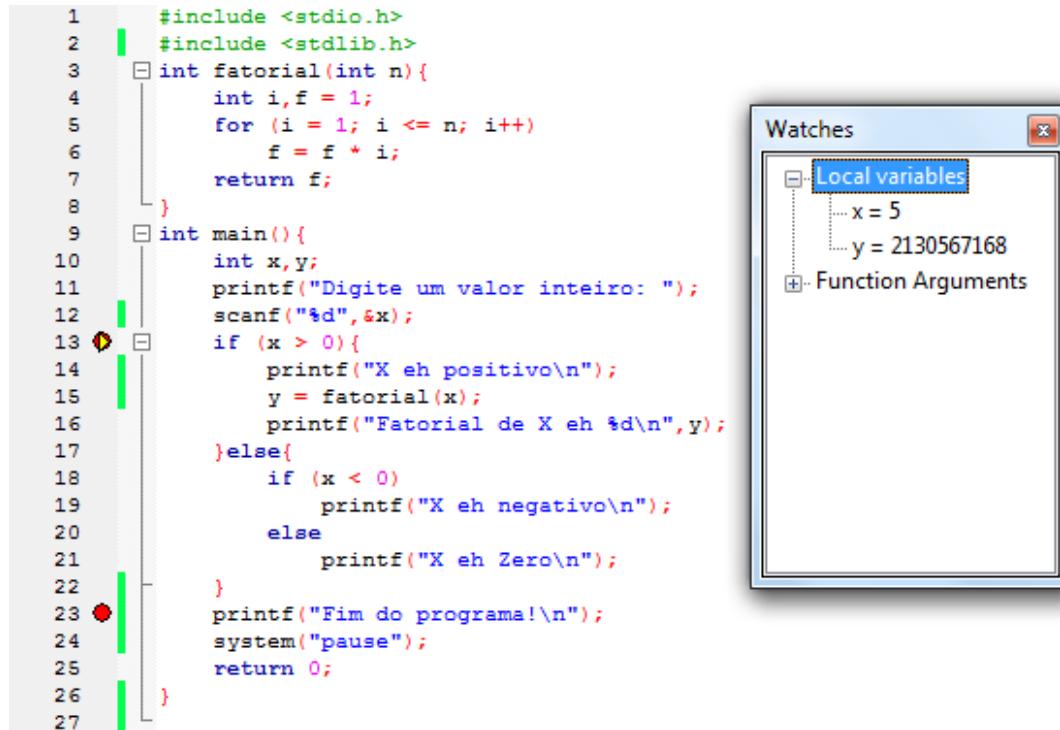
- No nosso exemplo, o usuário deverá digitar, no console, o valor lido pelo comando **scanf()** e depois retornar para a tela do **Code::Blocks** onde o programa se encontra pausado
  - Note que existe um **triângulo amarelo** dentro do primeiro **breakpoint**
  - Esse triângulo indica em que parte do programa a pausa está

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  int fatorial(int n){
4      int i,f = 1;
5      for (i = 1; i <= n; i++)
6          f = f * i;
7      return f;
8  }
9  int main(){
10     int x,y;
11     printf("Digite um valor inteiro: ");
12     scanf("%d",&x);
13     if (x > 0){
14         printf("X eh positivo\n");
15         y = fatorial(x);
16         printf("Fatorial de X eh %d\n",y);
17     }else{
18         if (x < 0)
19             printf("X eh negativo\n");
20         else
21             printf("X eh Zero\n");
22     }
23     printf("Fim do programa!\n");
24     system("pause");
25     return 0;
26 }
27 
```

# Utilizando o Debugger do Code::Blocks

- Dentro da opção **Debugging Windows**, podemos habilitar a opção **Watches**
  - Essa opção vai abrir uma pequena janela que permite ver o valor atual das variáveis de um programa, assim como o valor passado para funções



The image shows the Code::Blocks IDE interface. On the left is the code editor with a C program. On the right is the 'Watches' window.

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  int fatorial(int n){
4      int i,f = 1;
5      for (i = 1; i <= n; i++)
6          f = f * i;
7      return f;
8  }
9  int main(){
10     int x,y;
11     printf("Digite um valor inteiro: ");
12     scanf("%d",&x);
13     if (x > 0){
14         printf("X eh positivo\n");
15         y = fatorial(x);
16         printf("Fatorial de X eh %d\n",y);
17     }else{
18         if (x < 0)
19             printf("X eh negativo\n");
20         else
21             printf("X eh Zero\n");
22     }
23     printf("Fim do programa!\n");
24     system("pause");
25     return 0;
26 }
27
  
```

The 'Watches' window displays the current values of variables:

- Local variables**
  - x = 5
  - y = 120
- Function Arguments**

# Utilizando o Debugger do Code::Blocks

- A partir de determinado ponto de pausa do programa, podemos nos mover para a próxima linha do programa com a opção **Next line (F7)**
- Essa opção faz com que o programa seja executado passo a passo, sempre avançando para a linha seguinte do escopo onde estamos

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  int factorial(int n){
4      int i,f = 1;
5      for (i = 1; i <= n; i++)
6          f = f * i;
7      return f;
8  }
9  int main(){
10     int x,y;
11     printf("Digite um valor inteiro: ");
12     scanf("%d",&x);
13     if (x > 0){
14         printf("X eh positivo\n");
15         y = factorial(x);
16         printf("Fatorial de X eh %d\n",y);
17     }else{
18         if (x < 0)
19             printf("X eh negativo\n");
20         else
21             printf("X eh Zero\n");
22     }
23     printf("Fim do programa!\n");
24     system("pause");
25     return 0;
26 }
27

```

# Utilizando o Debugger do Code::Blocks

- Se houver uma chamada de função (linha 15) a opção **Next line (F7)** chama a função, mas não permite que a estudemos passo a passo
- Para entrar dentro do código de uma função, utilizamos a opção **Step into (Shift+F7)** na linha da chamada da função
- Nesse caso, o triângulo amarelo que marca onde estamos no código vai para a primeira linha do código da função

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  int factorial(int n){
4      int i,f = 1;
5      for (i = 1; i <= n; i++)
6          f = f * i;
7      return f;
8  }
9  int main(){
10     int x,y;
11     printf("Digite um valor inteiro: ");
12     scanf("%d",&x);
13     if (x > 0){
14         printf("X eh positivo\n");
15         y = factorial(x);
16         printf("Fatorial de X eh %d\n",y);
17     }else{
18         if (x < 0)
19             printf("X eh negativo\n");
20         else
21             printf("X eh Zero\n");
22     }
23     printf("Fim do programa!\n");
24     system("pause");
25     return 0;
26 }
27
  
```

# Utilizando o Debugger do Code::Blocks

- Uma vez dentro de uma função, podemos percorrê-la passo a passo com a opção **Next line (F7)**
  - Terminada a função, o **debugger** vai para a linha seguinte ao ponto do código que chamou a função (linha 16)
  - Caso queiramos ignorar o resto da função e voltar para onde estávamos no código que chamou a função, basta clicar na opção **Step out (Shift+Ctrl+F7)**

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 int factorial(int n){
4     int i,f = 1;
5     for (i = 1; i <= n; i++)
6         f = f * i;
7     return f;
8 }
9 int main(){
10    int x,y;
11    printf("Digite um valor inteiro: ");
12    scanf("%d",&x);
13    if (x > 0){
14        printf("X eh positivo\n");
15        y = factorial(x);
16        printf("Fatorial de X eh %d\n",y);
17    }else{
18        if (x < 0)
19            printf("X eh negativo\n");
20        else
21            printf("X eh Zero\n");
22    }
23    printf("Fim do programa!\n");
24    system("pause");
25    return 0;
26 }
27
  
```

# Utilizando o Debugger do Code::Blocks

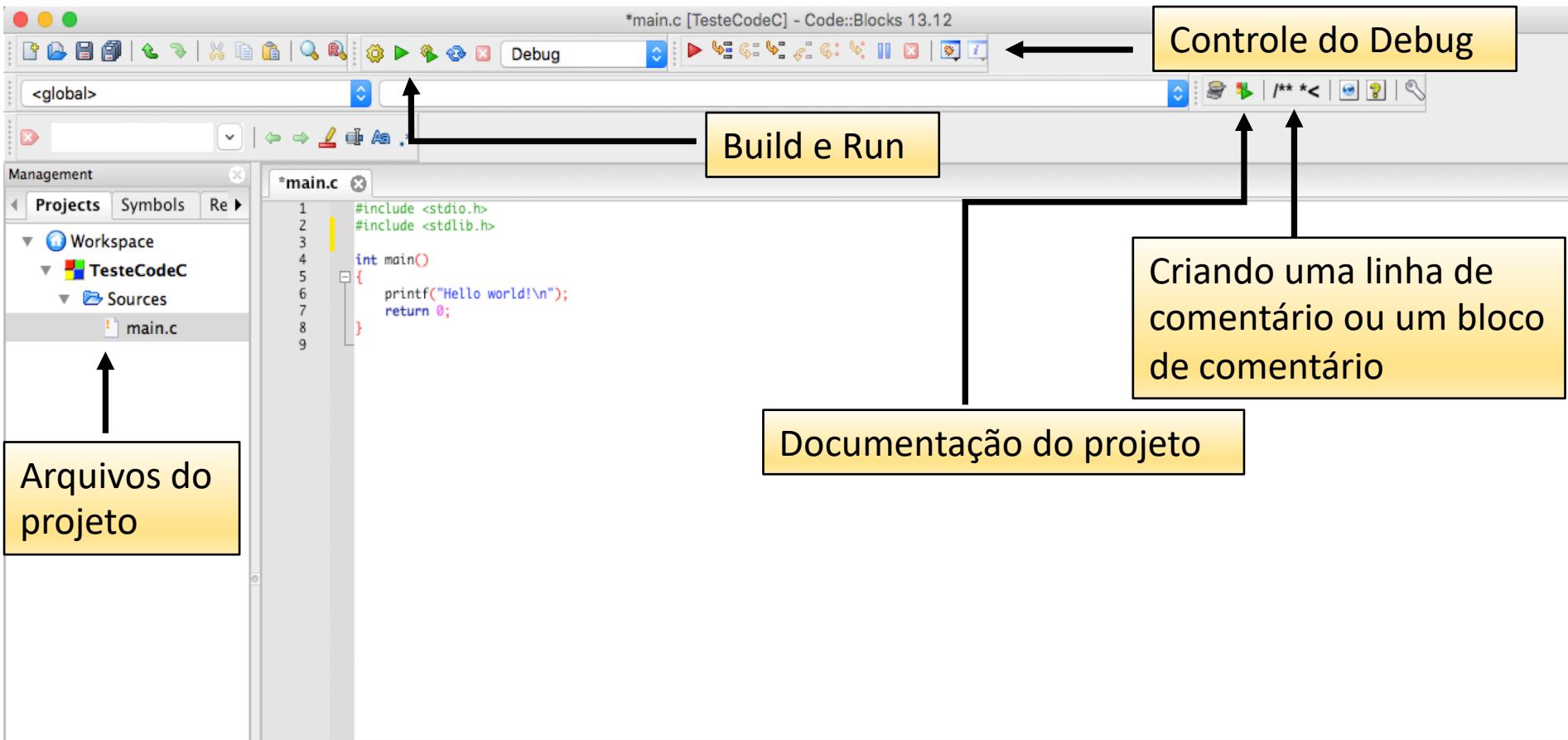
- Para avançar todo o código e ir direto para o próximo **breakpoint** (linha 23), podemos usar a opção **Continue (Ctrl+F7)**
- Por fim, para parar o **debugger**, basta clicar na opção **Stop debugger**

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  int fatorial(int n){
4      int i,f = 1;
5      for (i = 1; i <= n; i++)
6          f = f * i;
7      return f;
8  }
9  int main(){
10     int x,y;
11     printf("Digite um valor inteiro: ");
12     scanf("%d",&x);
13     if (x > 0){
14         printf("X eh positivo\n");
15         y = fatorial(x);
16         printf("Fatorial de X eh %d\n",y);
17     }else{
18         if (x < 0)
19             printf("X eh negativo\n");
20         else
21             printf("X eh Zero\n");
22     }
23     printf("Fim do programa!\n");
24     system("pause");
25     return 0;
26 }
27

```

# Outras Funcionalidades do Code::Blocks



# Dúvidas?



# NETBEANS

---

# NETBEANS

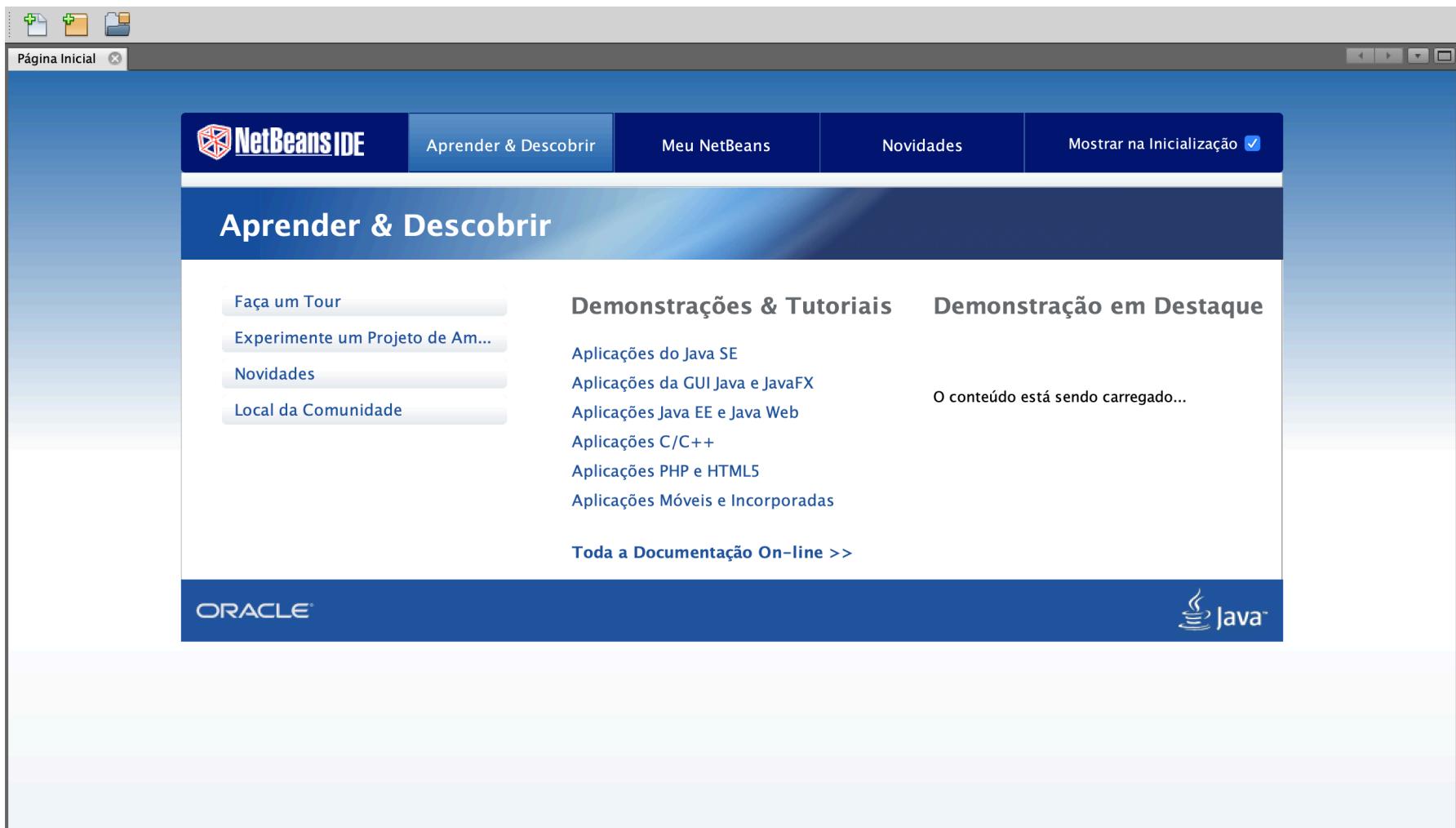
- NetBeans é uma IDE gratuita e de código aberto para desenvolvimento de software em diversas linguagens:
  - Java
  - HTML5
  - C/C++
  - PHP
  - ...

# NETBEANS

- O site do netbeans: <https://netbeans.org>
- O download pode ser feito pelo link:
  - [https://netbeans.org/downloads/8.0.1/?pagelang=pt\\_BR](https://netbeans.org/downloads/8.0.1/?pagelang=pt_BR)
  - Existe uma versão própria para desenvolvimento de aplicações C/C++
  - Porém, vamos utilizar a versão completa

Distribuições para baixar do NetBeans IDE					
Tecnologias suportadas *	Java SE	Java EE	C/C++	PHP	Tudo
SDK da plataforma NetBeans	•	•			•
Java SE	•	•			•
Java FX	•	•			•
Java EE		•			•
Java ME					—
HTML5		•		•	•
Java Card(tm) 3 Connected					—
C/C++			•		•
Groovy					•
PHP				•	•
Servidores embutidos					
GlassFish Server Open Source Edition 4.1		•			•
Apache Tomcat 8.0.9	•				•
<a href="#">Download</a> <a href="#">Download</a> <a href="#">Download</a> <a href="#">Download</a> <a href="#">Download</a>					
105 MB livre(s) 222 MB livre(s) 71 MB livre(s) 72 MB livre(s) 243 MB livre(s)					

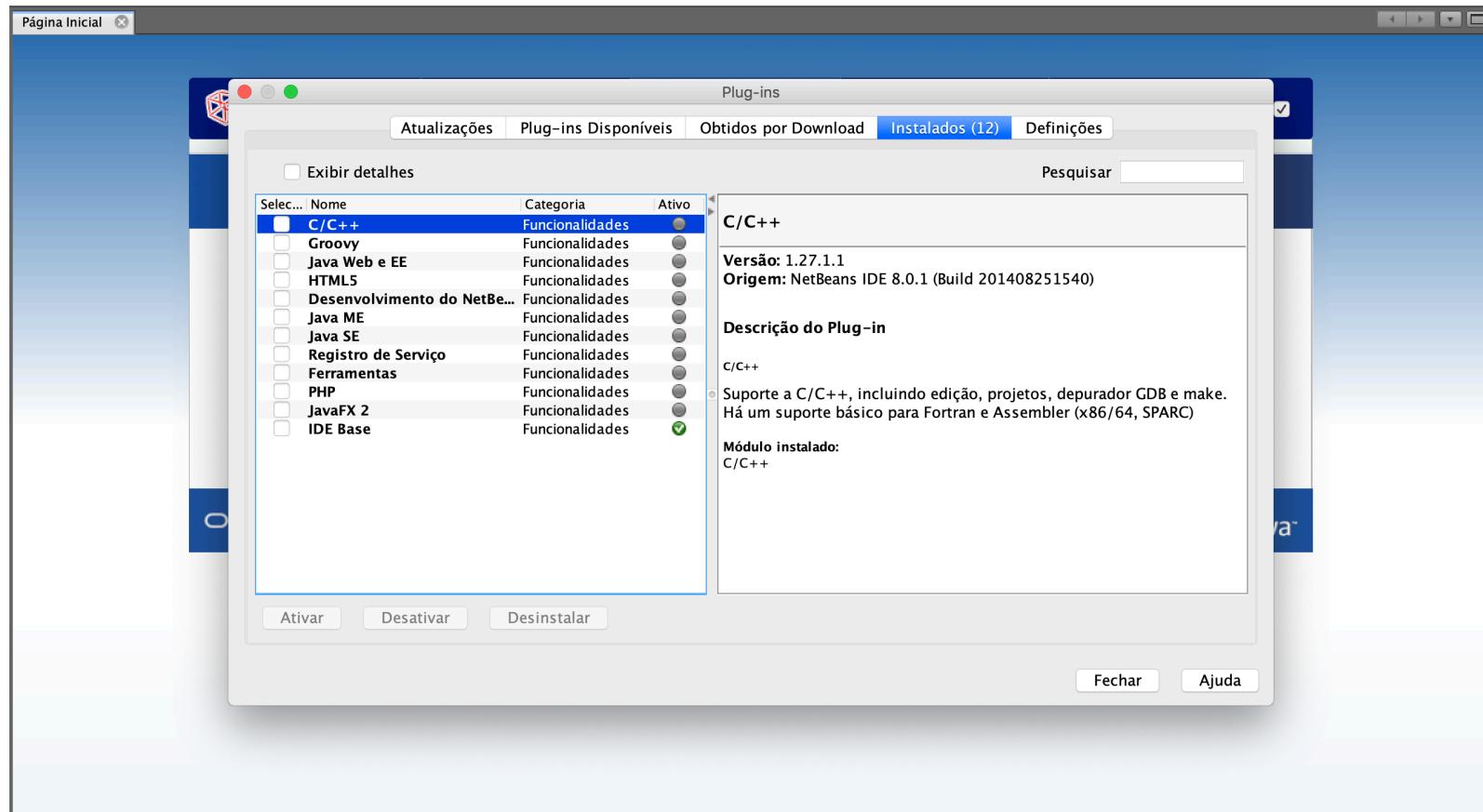
# NETBEANS



The screenshot shows the NetBeans IDE interface with a blue-themed header bar. The header includes icons for file operations (New, Open, Save), a search bar, and tabs for 'Página Inicial' and 'Aprender & Descobrir'. The main content area has a dark blue header with the title 'Aprender & Descobrir'. On the left, there's a sidebar with links: 'Faça um Tour', 'Experimente um Projeto de Am...', 'Novidades', and 'Local da Comunidade'. The central content area is divided into two columns: 'Demonstrações & Tutoriais' (with links to Java SE, GUI Java and JavaFX, Java EE and Java Web, C/C++, PHP and HTML5, and Mobile Applications) and 'Demonstração em Destaque' (with the message 'O conteúdo está sendo carregado...'). At the bottom, there are logos for 'ORACLE' and the 'Java' brand.

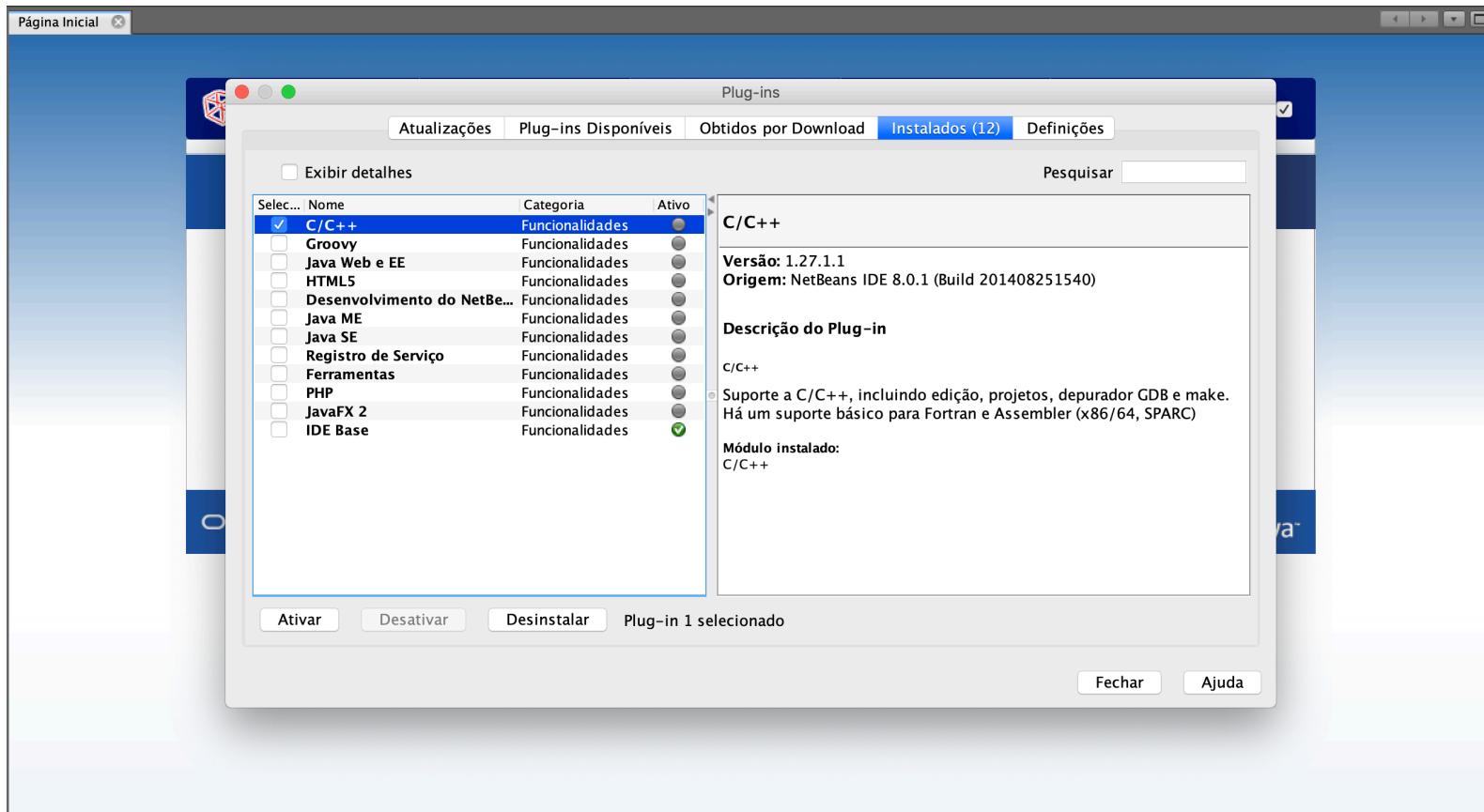
# NETBEANS

## Instalar Plug-ins → Instalados



# NETBEANS

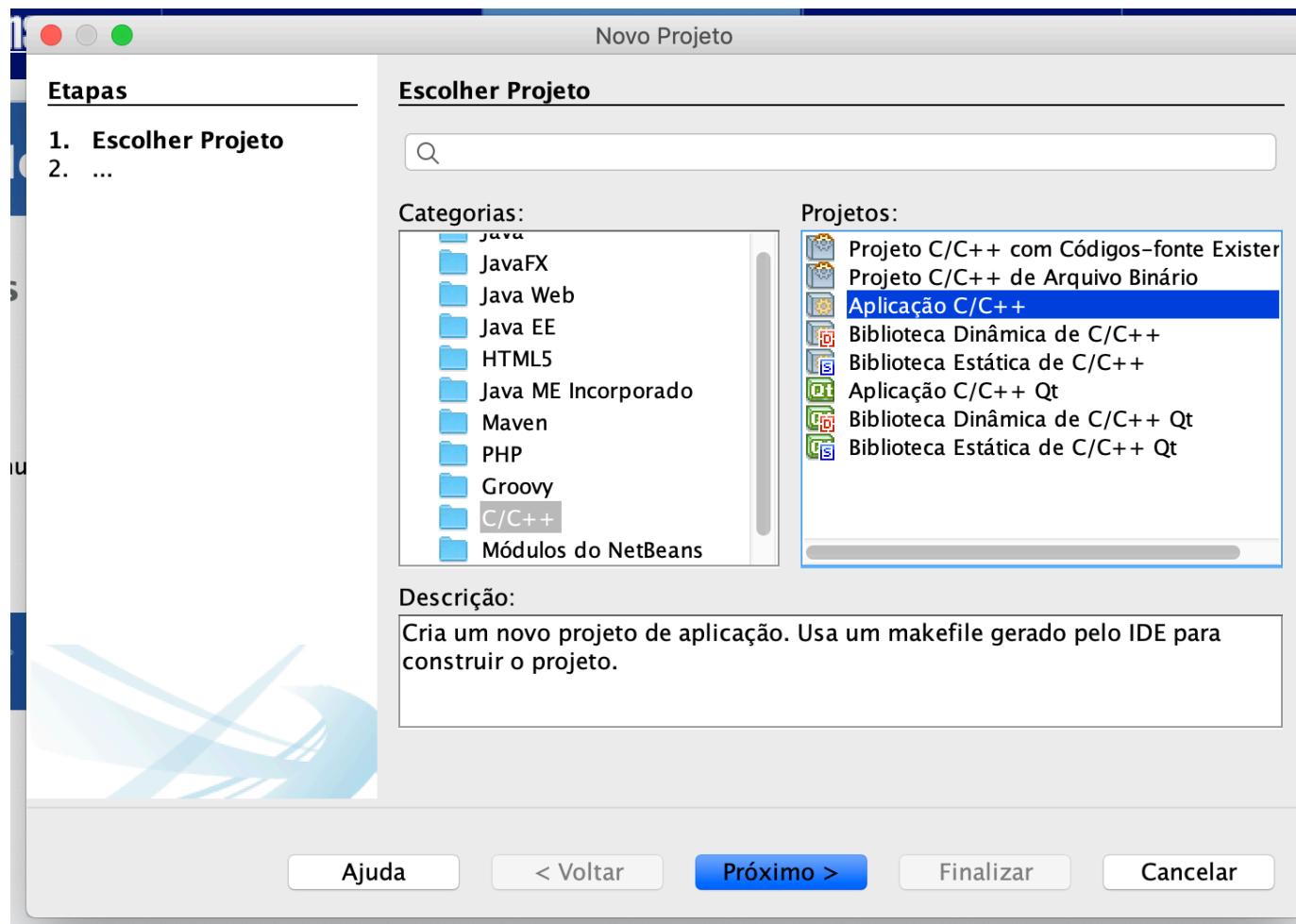
## Escolha: C/C++ → Ativar



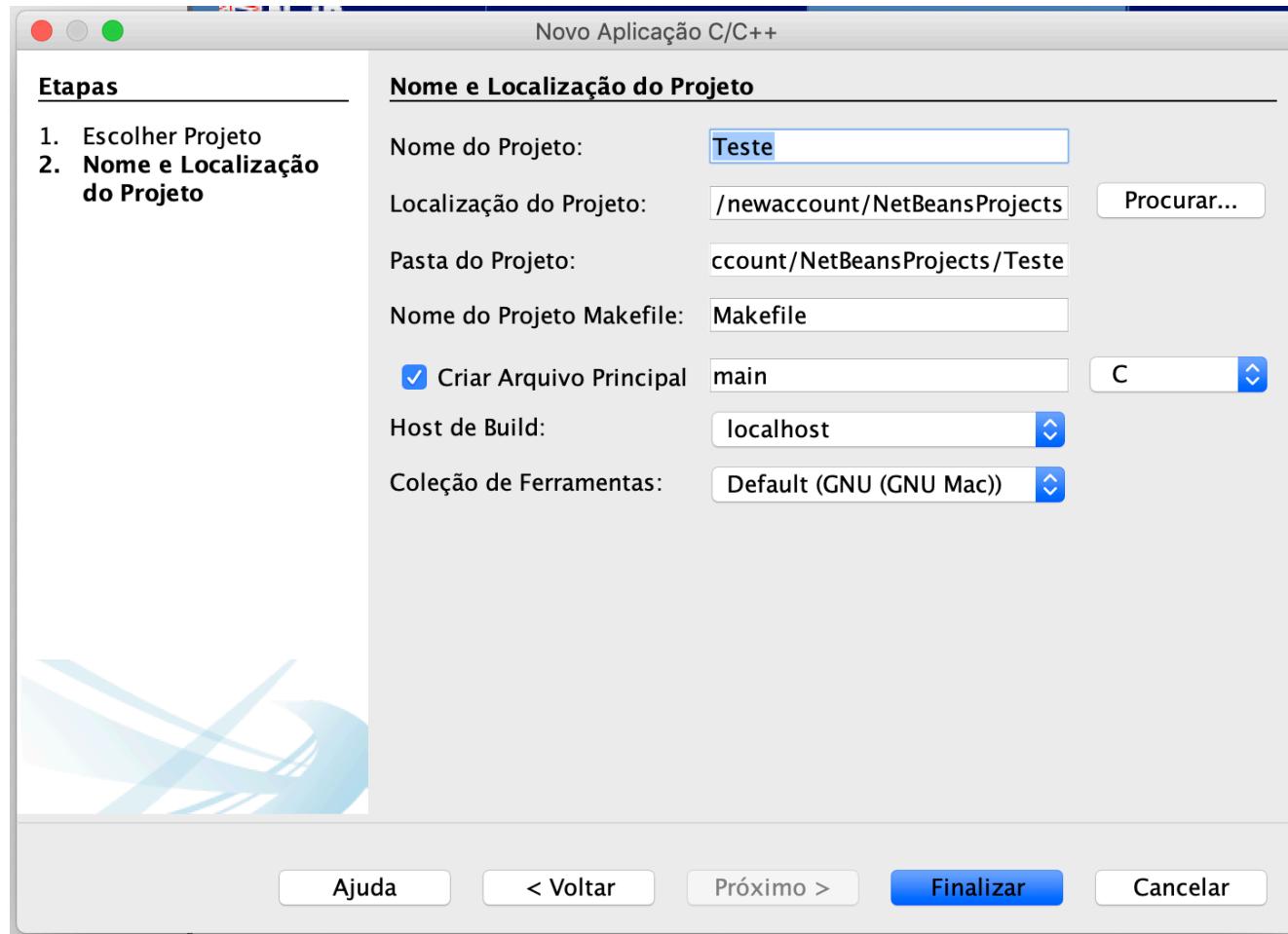
# NETBEANS

- Espere a instalação do Plug-in e pronto!
- Agora o NetBeans já está pronto para o desenvolvimento de aplicações C/C++
- Reinicie o NetBeans e crie um projeto

# NETBEANS



# NETBEANS



# NETBEANS

