Universidade de São Paulo – ICMC
Departamento de Ciências da Computação

SCC601 – Introdução a Ciência da Computação II

Professora: Rosane Minghim
Estagiária P.A.E: Nathalie Portugal

Manual Dev C++

Conteúdo

1. Introdução 1

2. Conhecendo o Dev-C++ 2

3. Barras de Navegação importantes: 5

a. Barra Executar: 5

b. Barra Depurar 7

4. Criação de Projetos 7

5. Usando o Depurador 14

6. Referencias **20**

1. Introdução

Atenção: O conteúdo do seguinte manual é uma tradução e adaptação de um manual feito em 2003. (Zheo,2003).

Um IDE é uma agrupação de ferramentas destinadas ao desenvolvimento, de forma que com um só programada podamos ter aceso a tudo o que necessitamos para a criação de aplicações.Dev C++ é um programa de software livre, o que implica que podemos aceder ao mesmo código fonte de forma gratuita, e o entorno é potente.

Observe-se que Dev-C++ não é um compilador, senão simplesmente um entorno gráfico para utilizar o verdadeiro compilador que é o MinGW (Minimalist GNU Windows). Esse compilador é também software livre baixo a licença de GNU. Este compilador é uma conversão do famoso compilador GCC dos sistemas GNU-Linux, e também pode ser utilizado como o original, por linha de comandos e com os mesmos argumentos. Si você quer mais informação do compilador, pode entrar nas paginas oficiais listadas embaixo.

- http://www.mingw.org/
- http://www.fsf.org/software/gcc/gcc.html

Para fazer uma descarga do programa, podemos entrar na pagina oficial e baixar a ultima versão:

- http://www.bloodshed.net/dev/devcpp.html

2. Conhecendo o Dev-C++

Um projeto é uma agrupação de todos os arquivos que requere uma aplicação, assim como de sua configuração especifica. Quando em Dev-C++ cria-se um novo projeto será criado um arquivo dev. É um arquivo que tem as referencias dos arquivos necessários para compilar nossa aplicação (código fonte, arquivos de cabeceira, etc.) assim como as opções necessárias especificas dessa aplicação (como podem ser referencias a bibliotecas não standard que precisam de compilação).

Também cumpre uma função similar à utilidade make de Linux, que revisa as dependências entre os arquivos que são parte do projeto para souber quais foram modificados (esses arquivos seriam os únicos que precisam compilação), e por isso é criado um arquivo chamado makefile.win. Então quando queremos criar uma aplicação distribuída em vários arquivos o que sejam dependentes de mais bibliotecas ou recursos é recomendável criar um projeto.

Logo da instalação do Dev-C++ teremos uma janela como a mostrada na seguinte figura:

Professora: Rosane Minghim
Estagiária P.A.E: Nathalie Portugal

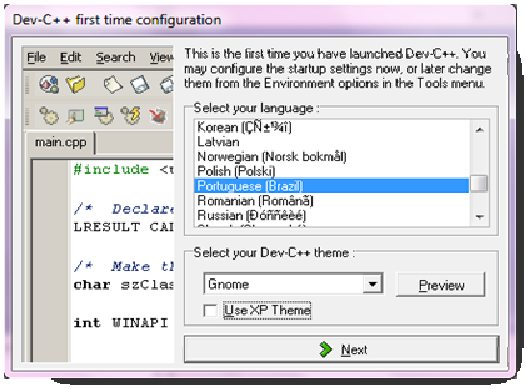


Figura 1 Configuração Inicial

Nesta janela podemos selecionar a linguagem que prefiramos e o tema que será aplicado em nosso programa, o tema só muda os ícones e as cores. Depois de configurar os parâmetros iniciais (por defeito só é pressionar seguinte até o final).

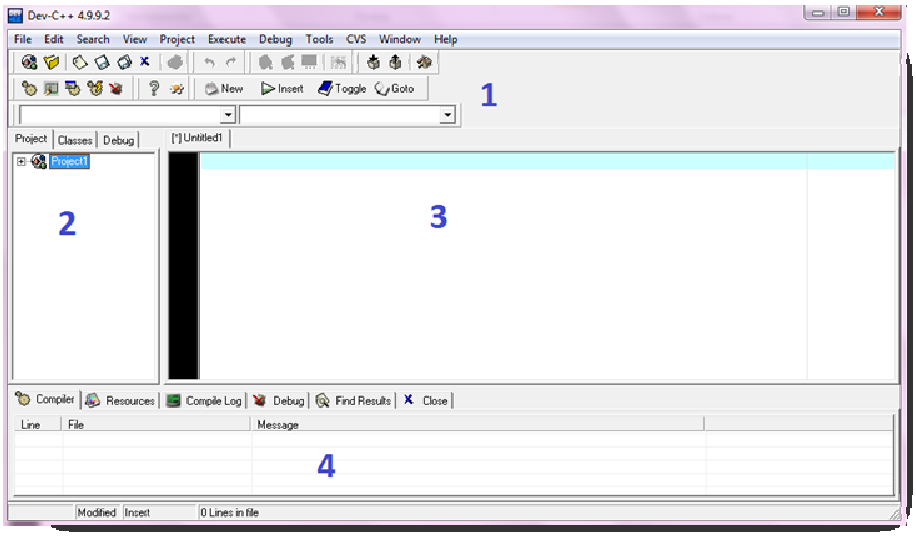


Figura 2 Entorno Dev-C++

Agora vamos descrever cada uma das áreas:

1) Barra de navegação e barra de ferramentas

Nesta parte estão as barras de navegação com os típicos comandos como abrir, guardar, copiar, etc. Além disso, também temos uma serie de ícones nas barras de ferramentas que são parte dos comandos mencionados acima. Em caso o efeito do ícone não esteja claro, pode-se deixar o ponteiro do mouse acima do ícone por uns segundos e uma descrição aparecera numa pequena janela amarela.

2) Explorador de projetos e classes, informação de depuração

Nesta área temos aceso a:

a. *Explorador de Projetos*, que mostra os arquivos pelos que esta formado nosso projeto, e por tanto nossa aplicação, já seja código, arquivos de cabeceira, ou recursos.

b. *Explorador de Classes* é uma das funções mais úteis, e podem se observar cada uma das estruturas ou classes definidas em nosso projeto, assim como os métodos e os dados que formam parte da estrutura ou classe, incluindo os argumentos e seu tipo.

   Também se pode observar as funções globais dentro do projeto, assim como seus parâmetros, Fazendo Double click num método ou função, iremos diretamente a onde foi implementada aquela função.

c. *Informação de depuração* aqui pode definir as variáveis que queríamos observar quando estamos depurando um programa.

---

3) Área de Edição

   Nesta área apareceram os arquivos de código que sejam abertos. Se pode ter mais de um arquivo aberto ao mesmo tempo, e pode ser selecionado pelas barras que estão acima.

4) Resultados da compilação e controles de depuração

   Quando selecionemos uma barra de acima, esta será expandida para mostrar-nos os resultados.

   Na barra Compilador olharemos os erros e as advertências que foi gerado durante a compilação de nosso código. Colocando um Double clik em algun deles Dev-C++ nos levara na linha onde foi gerado o erro.

   Na barra Resultados do compilador, teremos toda a salida que é gerada pelo compilador gcc, isso inclui também erros e notificações como na barra anterior, só que nesta barra si fazemos Double click não seremos levados na linha do erro.

3. Barras de Navegação importantes:

a. Barra Executar:

Professora: Rosane Minghim
Estagiária P.A.E: Nathalie Portugal

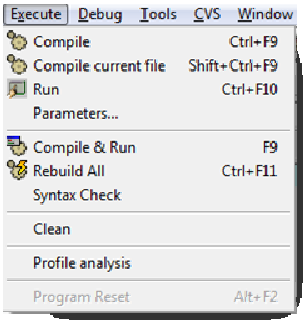


Figura 3 Barra Ejecutar

- Compilar: compila e enlaça o arquivo ou o projeto inteiro para criar o executável ou livraria.
- Compilar o arquivo atual compila o arquivo que esta sendo editado atualmente.
- Executar: executa o programa (só si existe o executável), é importante observar que esta função não compila o programa, então si alguma coisa foi modificada dentro do programa pero não é re compilado e pulsamos o botão executar, será executada a ultima versão compilado.
- Parâmetros: Si queremos que nosso executável receba algum tipo de parâmetro se pode colocar aça.
- Compilar e executar: Compila, enlaça e automaticamente executa o programa, se não houve nenhum erro.
- Reconstruir tudo: Apaga todos os arquivos de código objeto, para re compilar todos os arquivos de código fonte. Também enlaça todos, e reconstrói o projeto desde zero.
- Revisar sintaxe: Comprova si a sintaxe do código é correta.
- Limpar os resultados: Apaga todos os arquivos do código objeto e o executável, si existem.
- Análise de perfil: Mostra a informação do profiler, que consiste em dados estatísticos acerca das funções do programa, como o numero de vezes que são chamadas, o tempo que demoram em executasse, o percentagem de tempo total de programas que são executados, etc. Serve para encontrar as funções mais utilizadas ou as mais lentas, com a finalidade de aperfeiçoar-las. Ainda não funcionam bem estas opções.

---

b. Barra Depurar

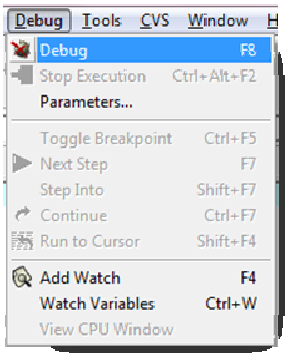


Figura 4 Depurar

- Depurar: Inicia a depuração do projeto.
- Para execução: Finaliza a execução do programa assim como o modo depuração.
- Adicionar ou apagar ponto de quebra (brakpoint): Adiciona um breakpoint, é disser um ponto no qual a execução do programa parará.
- Avançar passo a passo: Executa passo a passo as linhas dentro de um procedimento.
- Saltar passo: É como disser continuar, já que continua a execução do programa até que seja encontrado outro breakpoint ou finalize.
- Adicionar Watch: Abre um quadro de dialogo para introduzir a variável que queremos observar em tempo de execução. A variável e seu conteúdo aparecerá na área 2.
- Ver janela da CPU: É um debug avançado, onde pomos observar os registros da maquina no momento da execução.

4. Criação de Projetos

Para criar o novo projeto, vamos a basear-nos num projeto já criado que vem com os exemplos de Dev-c++ (quando se instala o dev, os exemplos também são

instalados no computador), para mostrar o uso do Dev-C++ vamos criar um projeto semelhante ao exemplo de Dev.

Usaremos os arquivos localizados em:

Si você que observar o projeto, pode fazer um Double click sobro um arquivo chamado FileEditor.dev, como é mostrado na figura:

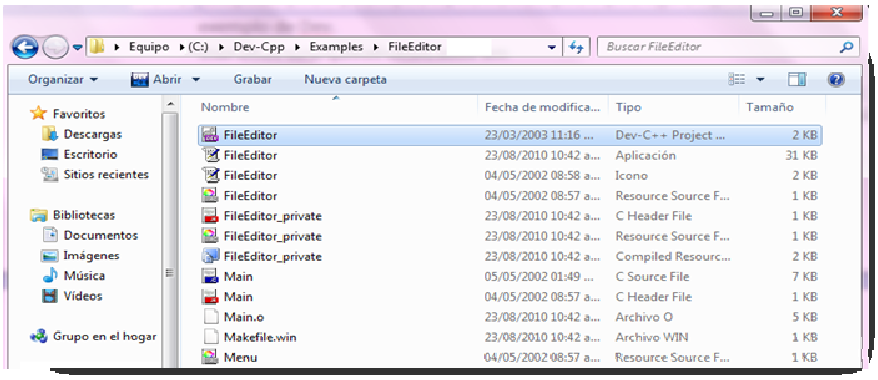


Figura 5 Diretório de Exemplos Dev-C++

Para nosso novo projeto, temos que criar uma pasta em algum lugar da computadora, ali devemos colocar os seguintes arquivos do diretório FileEditor, main.h, main.c, menu.rc.

Agora em Dev-C++ pulsamos “Novo Projeto” e deveria sair uma janela como a seguinte:

Professora: Rosane Minghim
Estagiária P.A.E: Nathalie Portugal

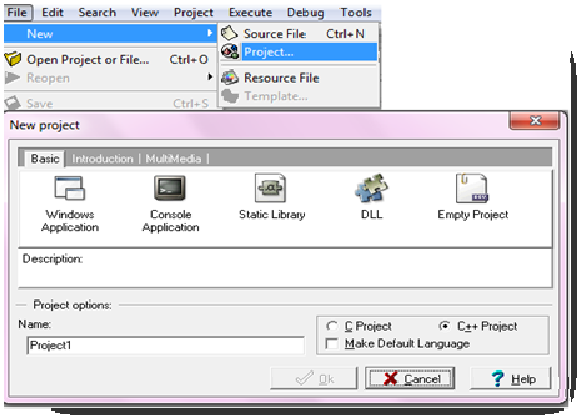


Figura 6 Novo Projeto

Seleciona a aplicação para windows (Windows application) e como nome de projeto "MyFileEditor" e assegure de selecionar "C Project" para que os arquivos tenham a extensão .c. Coloque aceitar e se mostrara uma janela para colocar o diretório onde guardar o arquivo do projeto, guarde-lo no mesmo diretório onde copio os arquivos anteriores.

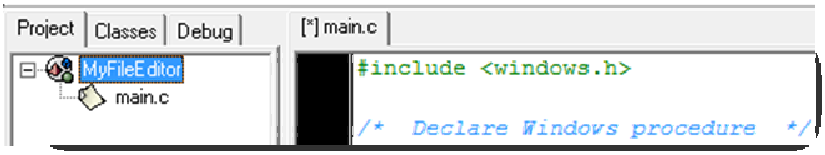


Figura 7 Projeto Adicionado

Como usaremos como base os arquivos que copiamos anteriormente, precisamos descartar o main.c que acabou de ser criado, para isso notamos que ao lado de "main.c" tem uma estrela *, isso significa que foram feitos algumas mudanças no arquivo mas ainda não foram salvadas. Como não precisamos o arquivo

simplesmente daremos click direto e fechar, e não damos click na opção de não salvar.

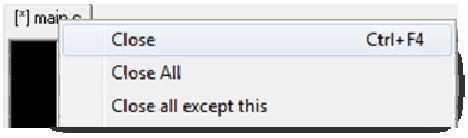


Figura 8 Fechar um arquivo

Agora temos que adicionar nossos próprios arquivos no projeto. Então vamos à barra de Projeto, click dereito sobre o ícone de MyFileEditor, depois coloque Adicionar ao projeto

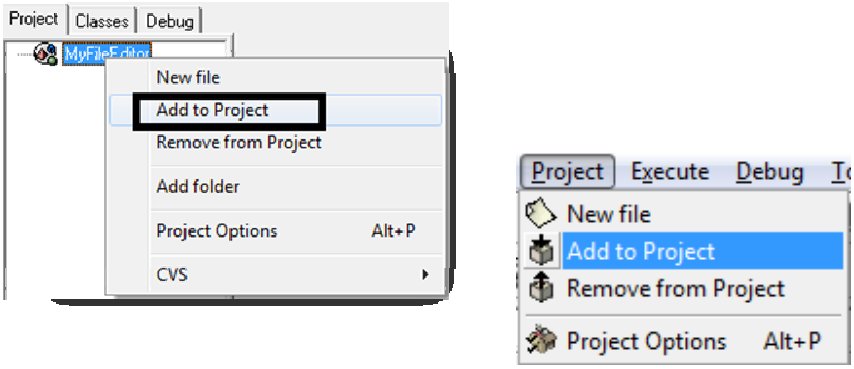


Figura 9 Adicionar ao projeto

No dialogo se apresentam os arquivos que copiamos dos exemplos de Dev, vamos adicioná-los a nosso projeto.

Professora: Rosane Minghim
Estagiária P.A.E: Nathalie Portugal

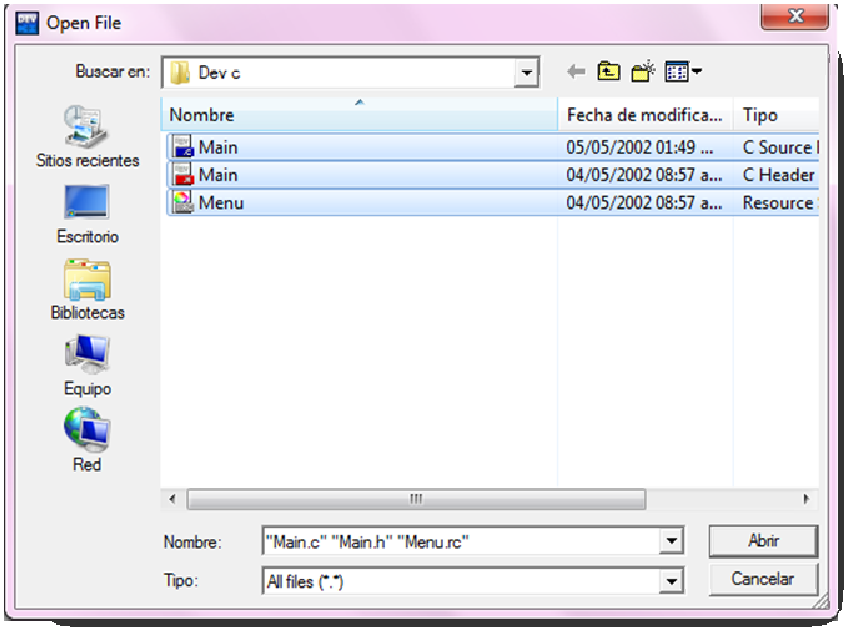


Figura 10 Adicionando elementos

Depois de adicionar os arquivos nossa barra de projetos deve estar como se mostra na figura abaixo:

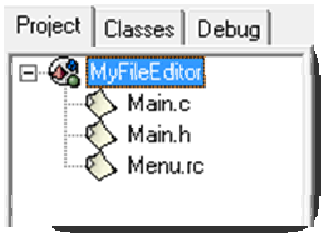


Figura 11 MyfileEditor

Mas imaginemos que nosso projeto tenha 12 arquivos de cabeceira (.h), 20 de código (.c ou .cpp) e 7 de recursos (.rc), isso ficaria um pouco desordenado. Mas Dev-C++ permite que possamos definir pastas para cada arquivo ou grupo de arquivos, para isso faça click direito no nome do projeto e selecione adicionar pasta e seleciona um nome. Neste caso criaremos 3 pastas, uma para o código, outra para a cabeceira e outro para os recursos, e depois arraste os itens para cada pasta.

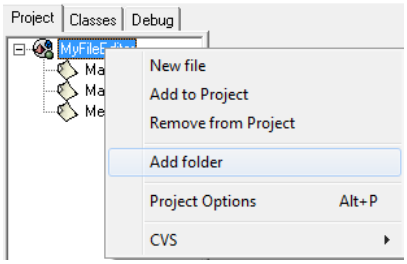


Figura 12 Adicionar Pasta

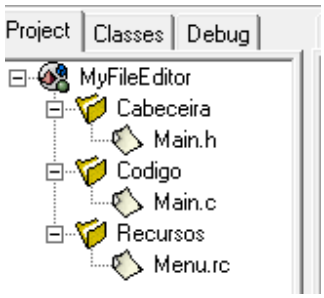


Figura 13 Pastas no Projeto

Agora compilamos o projeto, uma vez feito isso vamos ao diretório onde esta o projeto e podemos observar que temos mais arquivos que foram gerados, especialmente um executável, chamado MyFileEdit.exe, agora si o executamos observaremos um editor muito limitado.

Vamos trocar um pouco o projeto, para isso vamos a:

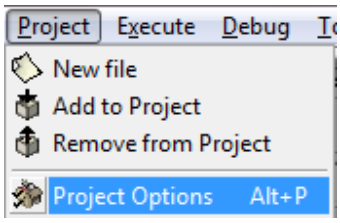


Figura 14 Project Options

Aparecera um dialogo com muitas barras, nos estamos na principal onde podemos definir o nome do projeto e seu tipo (Windows, consola, DLL,etc.), e também pomos selecionar o ícone, para isso pulsamos o botão livraria ou si nos temos algum ícone engraçado podemos colocar "browse". Coloque o ícone que você goste mais e a próxima vez que o projeto seja compilado aparecera com um ícone.

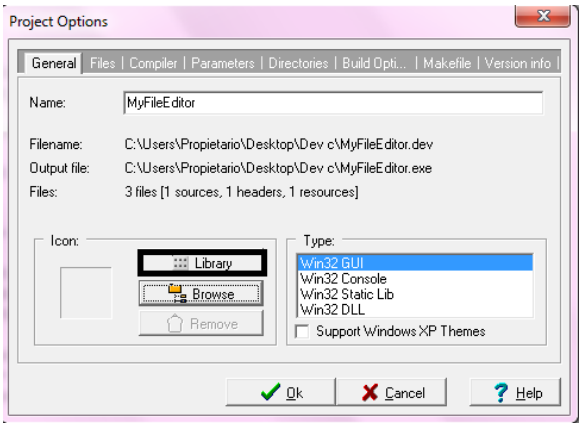


Figura 15 Selecionando um ícone

Na barra arquivos podemos nos referir a cada um dos arquivos de nosso projeto de forma individual, por exemplo, para definir qual deles tem prioridade na compilação, o si algum deles deve ser forçado a compilar como c++, o si queremos que algum deles não seja compilado. Isso só vai funcionar com arquivos de código já que são os únicos que são compilados.

Professora: Rosane Minghim
Estagiária P.A.E: Nathalie Portugal

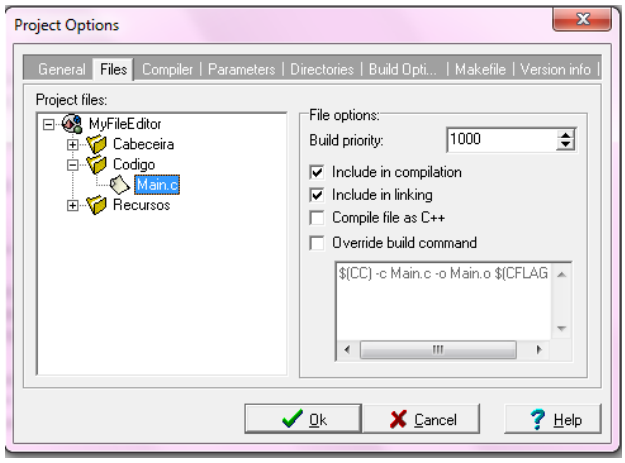


Figura 16 Menu Files

Na barra "Build" se pode especificar a pasta onde será gerado o executável uma vez compilado, enlaçado e gerado, a pasta onde se guarda o código objeto, uma vez compilados os arquivos, assim como o nome de arquivo que terá o executável.

Na barra "Makefile" nos pomos introduzir nossos próprios comandos makefile do projeto. Na barra "Version Info" pomos incluir certa informação sobre o projeto, como a versão do programa, e isso será incluído num arquivo de recursos gerado por Dev-C++.

5. Usando o Depurador

Para iniciar o depurador usaremos o mesmo projeto "MyFileEdit", mas primeiro temos que disser ao compilador que gere a informação necessária para que poda nos mostrar o programa e seus elementos em tempo de execução. Em outras palavras temos que fazer um executável com informação de depuração.
Para conseguir isso, abrimos o projeto e suas propriedades, e na barra "Compilador-linker" ativamos a opção de "gerar informação de debug". Também vamos nas opções de otimização e desativamos todas, apagamos também o argumento –s do linker na barra parâmetros e por ultimo reconstruímos todo o projeto

Professora: Rosane Minghim
Estagiária P.A.E: Nathalie Portugal

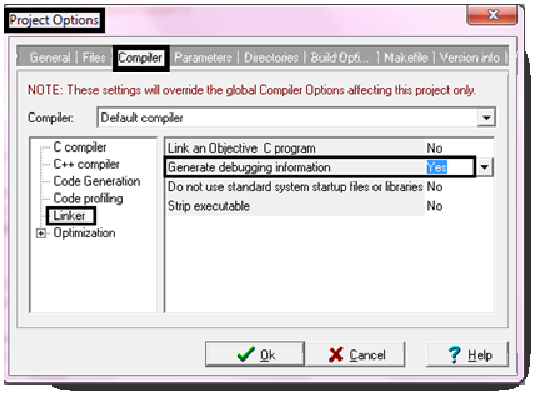


Figura 17 Project Options Parte 1

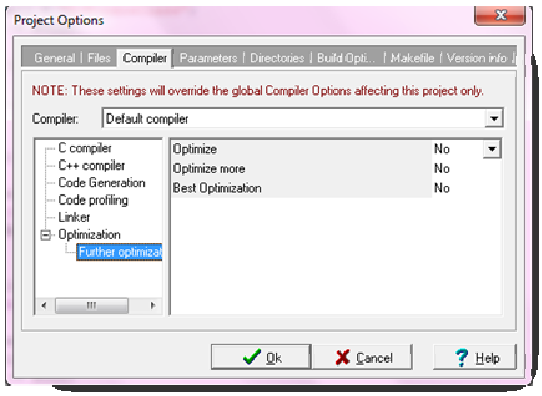


Figura 18 Project Options Parte 2

O tamanho do executável vai ser muito mais grande, por que toda a informação necessária para a depuração esta contida lá, por isso não é recomendável distribuir um programa com informação de depuração, por que alem de ocupar mais, é executado de uma maneira mais lenta.

Antes de começar a depurar, temos que colocar um breakpoint para disser ao depurador que quando chegue nesse lugar deve parar. Se não fazemos isso o programa será executado sem pausa nenhuma, o que não é útil. Por exemplo, abrimos o arquivo main.c, vamos na linha 180 (Dentro da função WinMain,

podemos usar o explorador de classes para procurar essa função) e coloca Ctrl+F5 (ou barra Depurar e depois estabelecer ponto de quebra), e deveria ficar assim:

Professora: Rosane Minghim
Estagiária P.A.E: Nathalie Portugal

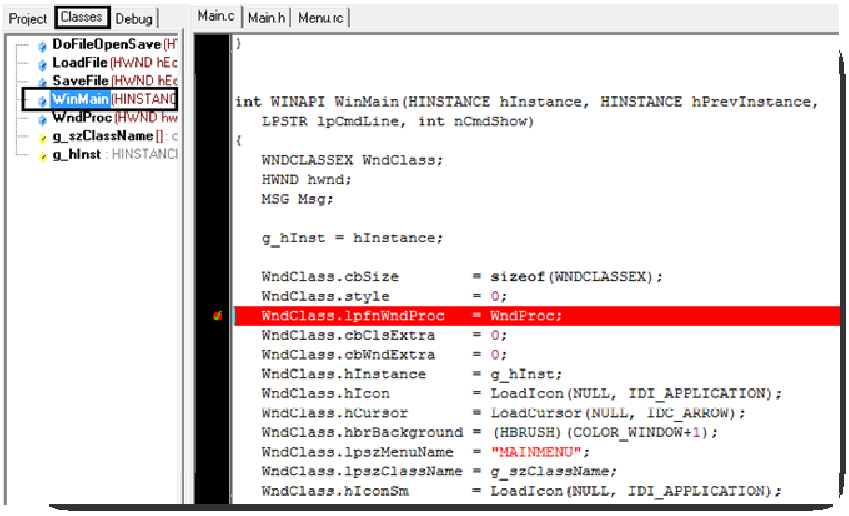


Figura 19 Função WinMain

Agora temos que pulsar F8 para começar a depuração. Você poderá observar que a linha troca de cor vermelho a azul e além tem uma seta azul na esquerda, isso significa que o ponto de execução do programa se encontra em essa linha, que é o mesmo que disser que a seguinte instrução que se executara será da linha azul. Disso se deduzi que quando se estabelece um breakpoint, o programa parará antes de executar a instrução da linha onde foi estabelecido.

Vamos observar as variáveis em tempo de execução, para isso pulsamos F4 ou pulsamos o botão “Adicionar Watch”.

Professora: Rosane Minghim
Estagiária P.A.E: Nathalie Portugal

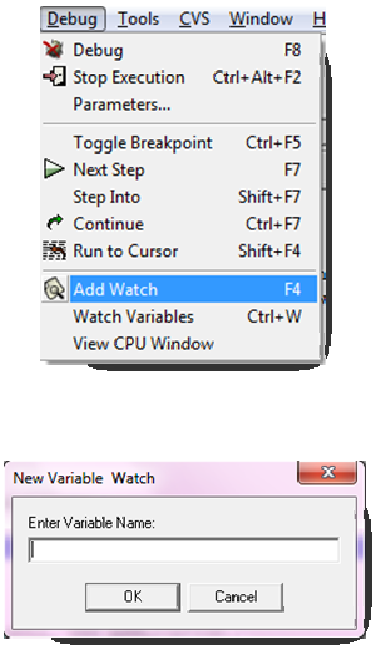
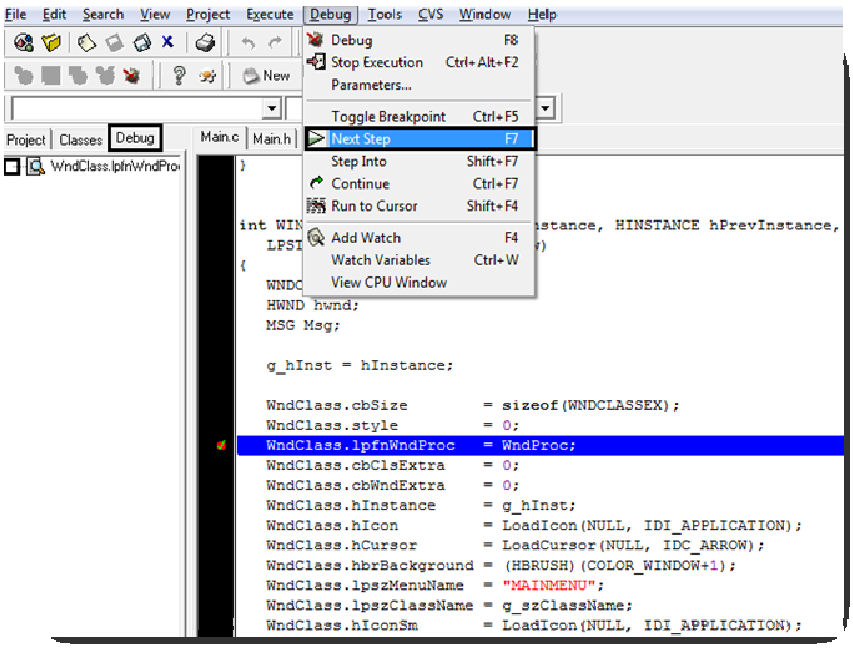


Figura 20 Observar uma variável.

Na janela mostrada na figura 20 introduzimos o seguinte texto sem as comas: "WndClass.lpfnWndProc", que é justo a variável que se inializara dentro de dois intruções (uma variável não inicializada aponta à memória, pelo que pode ter qualquer valor). Para olhar o valor da variável vamos executar o programa passo a passo, para isso pulsamos "Seguinte Passo".

Professora: Rosane Minghim
Estagiária P.A.E: Nathalie Portugal



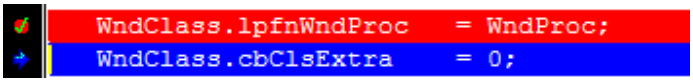


Figura 21 Executar passo a passo

Fazendo passo a passo podemos chegar a observar os valores que a variável selecionada vai tomando, podendo assim ser capazes de identificar um erro si houvesse.

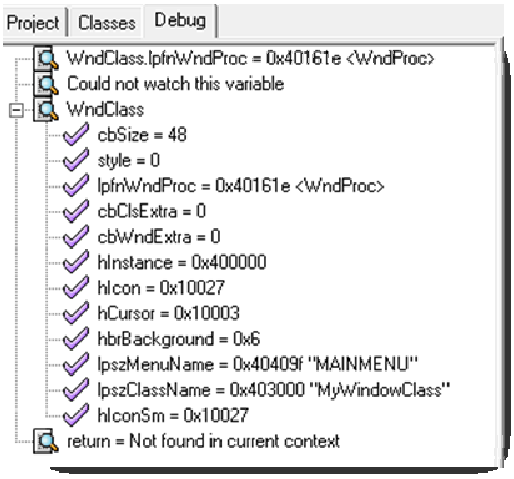


Figura 22 Barra Debug

A diferença entre os botões "seguinte passo" e "avançar passo a passo", é que o segundo permite se introduzir nas funções e o primeiro não. Por exemplo, a seguinte linha a executar é:

A=MyFunction()
B=5

Si colocamos "seguinte passo" passaremos na linha B=5, o código da função MyFunction() é executada pero não acedemos a ele e continuamos executando todo normalmente. Si colocamos "avançar passo a passo" o breakpoint se introduzi dentro de MyFunction() e vai mostrar passo a passo como se obtém os valores dentro dessa função.

6. Referencias

- (Zheo,2003). Dev-C++ Instalación y Primeros Pasos. Disponível em: http://www.scribd.com/doc/27470039/Dev-C-Instalacion-y-Primeros-Pasos. Ultima data de acceso: 16 de Agosto 2010.